Indice generale

Introduzione....9
1. Perché questa guida?....9
2. Chi dovrebbe leggere questo libro?....9
3. Nuove versioni e disponibilità....9
4. Storia delle revisioni....10
5. Contributi....11
6. Feedback....11
7. Informazioni sul copyright....11
8. Cosa vi serve?....12
9. Convenzioni utilizzate in questo documento....12
10. Organizzazione di questo documento....13
Capitolo 1. Cos'è Linux?....15
1.1. Storia....15
1.1.1. UNIX....15
1.1.2. Linus e Linux....16
1.1.3. Attuale utilizzazione dei sistemi Linux....17
1.2. L'interfaccia utente....18
1.2.1. Linux è difficile?....18
1.2.2. Linux per utenti non esperti....18
1.3. Linux ha futuro?....19
1.3.1. Open Source....19
1.3.2. Dieci anni di esperienza al vostro servizio....20
1.4. Caratteristiche di Linux....21
1.4.1. Vantaggi di Linux....21
1.4.2. Svantaggi di Linux....23
1.5. Sapore di Linux....24
1.5.1. Linux e GNU....24
1.5.2. GNU/Linux....25
1.5.3. Quale distribuzione dovrei installare?....25
1.6. Sommario....26
1.7. Esercizi....27
Capitolo 2. Avvio rapido....28
2.1. Connettersi, attivare l'interfaccia utente e disconnettersi....28
2.1.1. Introduzione....28
2.1.2. Modalità grafica....28
2.1.3. Modalità testo....30
2.2. Rudimenti essenziali....31
2.2.1. I comandi....31
2.2.2. Annotazioni generali....32
2.2.3. Usare le caratteristiche di Bash....33
2.3. Cercare aiuto....35
2.3.1. State attenti....35
2.3.2. Le pagine man....35
2.3.3. Maggiori informazioni....37

2.4. Sommario....40
2.5. Esercizi....41
2.5.1. Connessione e disconnessione....41
2.5.2. Password....42
2.5.3. Le directory....42
2.5.4. I file....43
2.5.5. Cercare aiuto....44
Capitolo 3. File e file system....45
3.1. Panoramica generale sul file system Linux....45
3.1.1. I file....45
3.1.2. Il partizionamento....47
3.1.3. Di più sulla struttura del file system....51
3.2. Orientarsi nel file system....54
3.2.1. Il percorso....54
3.2.2. Percorsi assoluti e relativi....56
3.2.3. I file e le directory più importanti....56
3.2.4. I file di configurazione più importanti....59
3.2.5. I più comuni device....61
3.2.6. I più comuni file di variabili....62
3.3. Manipolare i file....64
3.3.1. Vedere le proprietà dei file....64
3.3.2. Creare e cancellare file e directory....66
3.3.3. Trovare i file....70
3.3.4. Più modi di vedere il contenuto dei file....75
3.3.5. Collegare i file....76
3.4. La sicurezza dei file....78
3.4.1. Diritti di accesso: la prima linea di difesa di Linux....78
3.4.2. Gli strumenti....80
3.5. Sommario....86
3.6. Esercizi....87
3.6.1. Partizioni....88
3.6.2. Percorsi....88
3.6.3. Viaggio nel sistema....88
3.6.4. Manipolare i file....89
3.6.5. Permessi dei file....89
Capitolo 4. I processi....90
4.1. I processi in dettaglio....90
4.1.1. Multiutenza e multitasking....90
4.1.2. Tipi di processi....90
4.1.3. Attributi dei processi....93
4.1.4. Visualizzazione delle informazioni sui processi....93
4.1.5. Vita e morte di un processo....96
4.1.6. SUID e SGID....98
4.2. Processo d'avvio, init e shutdown....100
4.2.1. Introduzione....100
4.2.2. Il processo di avvio....101

4.2.3. Caratteristiche di GRUB......101
4.2.4. Init......102
4.2.5. I livelli di esecuzione di init......104
4.2.6. Lo spegnimento......106
4.3. La gestione dei processi......106
4.3.1. Lavori per l'amministratore di sistema......106
4.3.2. Quanto tempo richiede?......107
4.3.4. Le prestazioni......108
4.3.4. Il carico......108
4.3.5. Posso fare qualcosa come utente?......108
4.4. Temporizzare i processi......113
4.4.1. Usate quel tempo di ozio!......113
4.4.2. Il comando sleep......114
4.4.3. Il comando at......114
4.4.4. Cron e crontab......115
4.5. Sommario......117
4.6. Esercizi......118
4.6.1. In generale......118
4.6.2. Avvio, init, ecc......119
4.6.3. Pianificazione......119
Capitolo 5. Redirezione dell'I/O......120
5.1. Semplici redirezioni......120
5.1.1. Cosa sono lo standard input e lo standard output?......120
5.1.2. Gli operatori di redirezione......120
5.2. Caratteristiche avanzate della redirezione......123
5.2.1. Uso dei descrittori di file......123
5.2.2. Esempi......124
5.3. Filtri......125
5.3.1. Di più su grep......125
5.3.2. Filtraggio dei dati in uscita......126
5.4. Sommario......127
5.5. Esercizi......127
Capitolo 6. Gli editor di testo......129
6.1. Editor di testo......129
6.1.1. Perché dovrei usare un editor?......129
6.1.2. Quale editor dovrei usare?......129
6.2. Impiego dell'editor Vim......131
6.2.1. Due modi......131
6.2.2. Comandi di base......131
6.2.3. La maniera semplice......133
6.3. Linux in ufficio......133
6.3.1. Storia......133
6.3.2. Suite e programmi......134
6.3.3. Note......134
6.4. Sommario......135
6.5. Esercizi......135

Capitolo 7. Home sweet /home....136
7.1. Corretta gestione della casa in generale....136
7.1.1. Introduzione....136
7.1.2. Fare spazio....136
7.2. Il vostro ambiente testuale....140
7.2.1. Le variabili ambientali....140
7.2.2. I file di impostazione della shell....142
7.2.3. Un tipico insieme di file di configurazione....143
7.2.4. Il prompt di Bash....146
7.2.5. Gli script di shell....147
7.3. L'ambiente grafico....150
7.3.1. Introduzione....150
7.3.2. Il sistema X Window....150
7.3.3. Configurazione di un server X....153
7.4. Specifiche impostazioni di regione....153
7.4.1. Configurazione della tastiera....153
7.4.2. I tipi di caratteri....154
7.4.3. Data e fusi orari....154
7.4.4. La lingua....155
7.4.5. Specifiche informazioni nazionali....155
7.5. Installare nuovo software....156
7.5.1. In generale....156
7.5.2. I formati dei pacchetti....156
7.5.3. Gestione ed aggiornamenti automatici dei pacchetti....159
7.5.4. Aggiornare il kernel....161
7.5.5. Installare pacchetti extra dai CD di installazione....161
7.6. Sommario....163
7.7. Esercizi....164
7.7.1. L'ambiente della shell....164
7.7.2. L'ambiente grafico....164
Capitolo 8. Stampanti e stampe....166
8.1. I file di stampa....166
8.1.1. Stampare da linea di comando....166
8.1.2. Impostazione dei formati....168
8.2. Il lato server....169
8.2.1. In generale....169
8.2.2. Configurazione grafica della stampante....169
8.2.3. L'acquisto di una stampante per Linux....170
8.3. Problemi di stampa....170
8.3.1. File sbagliato....170
8.3.2. La mia stampa non è riuscita....170
8.4. Sommario....172
8.5. Esercizi....172
Capitolo 9. Tecniche fondamentali di backup....174
9.1. Introduzione....174
9.1.1. Preparazione dei vostri dati....174

9.2. Spostare i vostri dati verso un'unità di backup....179
9.2.1. Copiare su un disco floppy....179
9.2.2. Fare una copia con un masterizzatore di CD....180
9.2.3. Copie di sicurezza su/da unità jazz, periferiche USB e simili....181
9.2.4. Copie di sicurezza con una periferica a nastro....182
9.2.5. Strumenti dalla vostra distribuzione....182
9.3. Uso di rsync....183
9.3.1. Introduzione....183
9.3.2. Un esempio: rsync su una periferica USB di massa....183
9.4. Crittografia....183
9.4.1. Note generali....183
9.4.2. La generazione di una chiave....184
9.4.3. A proposito della vostra chiave....185
9.4.4. Crittografia dei dati....186
9.4.5. Decodifica dei file....186
9.5. Sommario....186
9.6. Esercizi....187
Capitolo 10. Le reti....188
10.1. Panoramica sulle reti....188
10.1.1. Il modello OSI....188
10.1.2. Alcuni popolari protocolli di rete....189
10.2. Configurazioni ed informazioni di rete....192
10.2.3. I comandi di configurazione delle reti....193
10.2.4. Nomi delle interfacce di rete....195
10.2.5. La configurazione del vostro host....196
10.2.6. Altri host....196
10.3. Applicazioni Internet/Intranet....198
10.3.1. Tipi di server....199
10.3.2. La posta....200
10.3.3. Il web....202
10.3.4. File Transfer Protocol....203
10.3.5. Chat e conferenze....204
10.3.6. Servizi per notizie....205
10.3.7. Il Domain Name System....206
10.3.8. DHCP....206
10.3.9. Servizi di autenticazione....206
10.4. Esecuzione remota di applicazioni....209
10.4.1. Introduzione....209
10.4.2. Rsh, rlogin e telnet....209
10.4.3. Il sistema X Window....210
10.4.4. La suite SSH....211
10.4.5. VNC....215
10.4.6. Il protocollo rdesktop....215
10.4.7. Cygwin....216
10.5. La sicurezza....216
10.5.1. Introduzione....216

10.5.2. I servizi....217
10.5.3. Aggiornare con regolarità....217
10.5.4. I firewall e le politiche d'accesso....218
10.5.5. La scoperta delle intrusioni....219
10.5.6. Ulteriori spunti....220
10.5.7. Sono stato attaccato dagli hacker?....220
10.5.8. Ripristinare dopo un'intrusione....221
10.6. Sommario....221
10.7. Esercizi....222
10.7.1. Le reti in generale....222
10.7.2. Connessioni remote....223
10.7.3. La sicurezza....223
Capitolo 11. Suoni e video....224
11.1. Le basi dell'audio....224
11.1.1. Installazione....224
11.1.2. I driver e l'architettura....224
11.2. Riproduzione audio e video....225
11.2.1. Ascolto e copia dei CD....225
11.2.2. La riproduzione di file musicali....225
11.2.3. La registrazione....227
11.3. Riproduzione video, guardare flussi e televisione....228
11.4. Telefonia Internet....229
11.4.1. Che cos'è?....229
11.4.2. Cosa vi serve?....229
11.5. Sommario....230
11.6. Esercizi....231
Appendice A. Dove andare da qui?....232
A.1. Libri utili....232
A.1.1. Linux in generale....232
A.1.2. Editor....232
A.1.3. Shell....232
A.1.4. X Window....232
A.1.5. Reti....233
A.2. Siti utili....233
A.2.1. Informazioni generali....233
A.2.2. Riferimenti a specifiche architetture....233
A.2.3 Distribuzioni....233
A.2.4. Software....234
Appendice B. Comandi DOS contro Linux....235
Appendice C. Caratteristiche della shell....236
C.1. Caratteristiche comuni....236
C.2. Caratteristiche diverse....237
Appendice D. GNU Free Documentation License....240
D.1. PREAMBLE....240
D.2. APPLICABILITY AND DEFINITIONS....240
D.3. VERBATIM COPYING....241

D.4. COPYING IN QUANTITY....242
D.5. MODIFICATIONS....242
D.6. COMBINING DOCUMENTS....244
D.7. COLLECTIONS OF DOCUMENTS....244
D.8. AGGREGATION WITH INDEPENDENT WORKS....245
D.9. TRANSLATION....245
D.10. TERMINATION....245
D.11. FUTURE REVISIONS OF THIS LICENSE....245
D.12. ADDENDUM: How to use this License for your documents....246
Glossario....247
A....247
B....248
C....248
D....249
E....250
F....251
G....251
H....252
I....253
J....253
K....254
L....254
M....255
N....256
O....257
P....257
Q....258
R....259
S....259
T....261
U....262
V....263
W....263
X....264
Y....265
Z....265

Introduzione

1. Perché questa guida?

Molte persone credono ancora che sia difficoltoso imparare Linux o che solo degli esperti possano comprendere come funziona un sistema Linux.
Sebbene sia disponibile una ricca documentazione gratuita, questa è largamente dispersa in rete e spesso confonde in quanto abitualmente destinata ad esperti UNIX o Linux. Oggi, grazie ai progressi nel suo sviluppo, Linux è cresciuto in popolarità sia in ambito domestico che lavorativo.
L'obbiettivo di questa guida è mostrare alla gente di tutte le età che Linux può essere semplice, divertente ed utilizzabile per ogni genere di impiego.

---

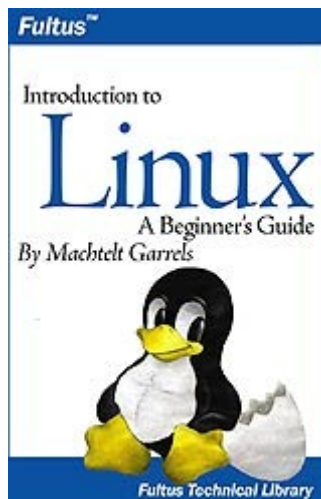
2. Chi dovrebbe leggere questo libro?

Questa guida è stata concepita come panoramica sul sistema operativo Linux, rivolta ai nuovi utenti alla stregua di un giro esplorativo e come guida introduttiva, con esercizi alla fine di ciascun capitolo: gli utenti più avanzati possono considerarla come una guida di riferimento e raccolta di nozioni basilari per l'attività di amministrazione di sistema e di rete. Questo libro contiene molti esempi reali derivati dall'esperienza dell'autrice come amministratrice di sistema Linux e di rete, insegnante e consulente. Noi speriamo che questi esempi vi aiuteranno ad avere una migliore comprensione del sistema Linux e che vi sentirete incoraggiati a condurre esperimenti in proprio.
Chi desidera ottenere una "CLUE", una Esperienza di Utente di Linea di Comando [ndt. Command Line User Experience] con Linux (e Unix in generale) troverà utile questo libro.

---

3. Nuove versioni e disponibilità

Questo documento è pubblicato nella sezione "Guide" della collezione del Progetto di Documentazione Linux (LDP = Linux Documentation Project) all'indirizzo: http://www.tldp.org/guides.html
Potete inoltre scaricare le versioni in PDF e PostScript.
La versione più recente è disponibile sul sito http://tille.garrels.be/training/tldp.
La guida si può far stampare su carta presso Fultus.com Books.
Fultus distribuisce questo documento presso molte librerie, comprese Baker & Taylor e quelle online Amazon.com, Amazon.co.uk, BarnesAndNoble.com e Google's Froogle.

Figura 1. Copertina di "Introduction to Linux"

La guida è stata tradotta in Hindi da:

- Alok Kumar
- Dhananjay Sharma
- Kapil
- Puneet Goel
- Ravikant Yuyutsu

Andrea Montagner ha tradotto la guida in italiano.

---

4. Storia delle revisioni

Storia delle revisioni

Revisione 1.25 20070511 Rivisto da: MG
Commenti dai lettori, aggiornamenti minori. Etichetta della posta elettronica, aggiornamento informazioni sulla disponibilità (grazie Oleg)

Revisione 1.24 2006-11-01 Rivisto da: MG
Aggiunti termini dell'indice, preparato per l'edizione di seconda stampa, aggiunte informazioni su gpg e proxy.

Revisione 1.23 2006-07-25 Rivisto da: MG e FK
Aggiornamenti e correzioni, rimossa nuovamente l'app5, adattata la licenza per consentire l'inclusione tra i documenti Debian.

Revisione 1.22 2006-04-06 Rivisto da: MG
Rivisto completamente il capitolo 8, capitolo 10: esempi chiarificatori, aggiunte informazioni su ifconfig e cygwin, riviste le applicazioni di rete.

Revisione 1.21 2006-03-14 Rivisto da: MG
Aggiunti esercizi nel capitolo 11, corretti errori di nuova riga, panoramica dei comandi completata per il capitolo 9, correzioni minori nel capitolo 10.

Revisione 1.20 2006-01-06 Rivisto da: MG
Diviso il capitolo 7: ora la materia dell'audio si trova in un capitolo separato, capitolo 11.xml. Piccole revisioni, aggiornamenti sui comandi come aptitude, di più sulle memorie USB, telefonia Internet, correzioni dai lettori.

Revisione 1.13 2004-04-27 Rivisto da: MG
Ultima rilettura prima di inviare il tutto a Fultus per le stampe. Aggiunto riferimento a Fultus nella sezione Nuove Versioni, aggiornate le sezioni Convenzioni e Organizzazione. Cambi minori nei capitoli 4, 5, 6 e 8, aggiunte informazioni su rdesktop nel cap. 10, aggiornato il glossario, rimpiazzati riferimenti a fileutils con coreutils, ringraziamenti ai traduttori in Hindi.

5. Contributi

Molte grazie alle persone che hanno condiviso le loro esperienze e, in particolare, agli utenti Linux del Belgio per avermi ascoltato fino in fondo ogni giorno sempre generosi nei loro commenti.

Pure un pensiero speciale a Tabatha Marshall per aver fatto sia una revisione realmente minuziosa, sia il controllo ortografico e stilistico, e a Eugene Crosser per aver individuato gli errori che noi due avevamo trascurato.

E grazie a tutti i lettori che mi hanno avvisato degli argomenti mancanti ed a chi ha aiutato a togliere gli ultimi errori, le definizioni e i caratteri poco chiari attraverso le difficoltà di scrivermi tutte le loro segnalazioni. Quelle sono le persone che mi hanno aiutato a mantenere aggiornata questa guida, come Filipus Klutiero, che ha effettuato una revisione completa nel 2005 e nel 2006 e mi ha aiutata ad inserire la guida nella collezione dei documenti Debian, ed Alexey Eremenko, che mi ha inviato la base per il capitolo 11.

Nel 2006 Suresh Rajashekara ha creato un pacchetto Debian di questa documentazione.

Infine, un grosso ringraziamento ai volontari che stanno attualmente traducendo questo documento in francese, svedese, tedesco, farsi, hindi e altre lingue ancora. E' un grosso lavoro che non va disprezzato: ammiro il vostro coraggio.

6. Feedback

Informazioni perdute, collegamenti perduti, caratteri perduti? Scrivete per posta elettronica al manutentore di questo documento:

<tille wants no spam _at_ garrels dot be>

Non scordatevi di controllare prima l'ultima versione!

7. Informazioni sul copyright

---

8. Cosa vi serve?

Vi servono un computer e un supporto contenente una distribuzione Linux. La maggior parte di questa guida è applicabile a tutte le distribuzioni Linux e ad UNIX in generale. A parte il tempo non ci sono altre richieste specifiche.

Lo “Installation HOWTO” contiene utili informazioni su come ottenere software Linux ed installarlo sui vostri computer: tratta anche delle richieste hardware e della coesistenza con altri sistemi operativi.

Immagini CD possono essere scaricate da linux-iso.com ed in altri posti, vedi Appendice A.

Un'interessante alternativa per quelli che non osano installare Linux nelle loro macchine è costituita dalle distribuzioni Linux che si possono avviare da un CD, come quella Knoppix.

---

9. Convenzioni utilizzate in questo documento

Sono presenti in questo testo le seguenti convenzioni tipografiche e di utilizzo:

Tavola 1. Convenzioni tipografiche e di utilizzo

Tipo di testo	Significato
“Testo tra virgolette”	Citazioni da persone, output di computer citato.
`Vista da terminale`	Input e output testuale di computer catturato dal terminale, di solito reso con uno sfondo grigio chiaro.
Comando	Nome di un comando che può essere inserito nella linea di comando.
`VARIABILE`	Nome di una variabile o di un puntatore al contenuto di una variabile, come `$VARNAME`.
`opzione`	Opzione di un comando come l'opzione `-a` nel comando **ls**

Tipo di testo	Significato
argomento	Argomento di un comando come in “leggere **man** *ls*”
prompt	Prompt dell'utente abitualmente seguito da un comando che voi battete in una finestra di terminale come hilda@home > **ls** -l
Comando opzioni *argomenti*	Sintassi o uso generico di un comando su linea separata.
filename	Nome di file o directory. Ad es. “Portarsi alla directory /usr/bin”.
Tasto	Tasto da premere sulla tastiera come, ad es. “battere **Q** per terminare”.
Bottone	Bottone grafico da premere come, ad es. il pulsante OK.
Menu->Scelta	Scelta da effettuare in un menu grafico, per es. “Scegli Aiuto->Informazioni su Mozilla nel vostro browser”.
Terminologia	Termine o concetto importante: ”Il *kernel* Linux è il cuore del sistema”.
\	La sbarra inversa in una vista da terminale o in un sommario di comandi indica una linea non terminata. In altre parole se vedete un comando lungo che è tagliato in più linee, \ significa “Non premere ancora **Invio**!”.
Vedi Capitolo 1	Collegamento al relativo soggetto della guida
L'autrice	Collegamento attivo ad una risorsa di rete esterna.

Vengono utilizzate nel testo le seguenti immagini:

10. Organizzazione di questo documento

Questa guida fa parte del Progetto di Documentazione Linux (TLDP) e mira a costituire la base per tutti i materiali ivi rintracciabili. Come tale fornisce la conoscenza elementare necessaria a chi vuole iniziare a lavorare con Linux ed allo stesso tempo evita intenzionalmente di reinventare l'acqua calda. Così potete attendervi che questo libro sia incompleto e pieno di riferimenti a fonti di informazioni aggiuntive sul vostro sistema, su internet e nella vostra documentazione di sistema.

Il primo capitolo è un'introduzione al soggetto Linux; i due successivi trattano i comandi assolutamente di base. I capitoli 4 e 5 spiegano alcuni argomenti più avanzati ma sempre di base. Il capitolo 6 serve per procedere con il resto in quanto parla della modifica dei file, una capacità che bisogna acquisire per passare da principiante ad utente Linux. I successivi capitoli illustrano qualche altro argomento più sofisticato con cui avrete a che fare nell'uso quotidiano di Linux.

Tutti i capitoli sono forniti di esercizi che controlleranno la vostra preparazione per il capitolo successivo.

- Capitolo 1: Cos'è Linux, come è nato, vantaggi e svantaggi, cosa riserva il futuro per Linux, chi dovrebbe usarlo, installazione sul vostro computer.
- Capitolo 2: Iniziare, connettersi al sistema, comandi base, dove trovare aiuto.
- Capitolo 3: Il filesystem, directory e file importanti, gestione di file e directory, protezione dei vostri dati.
- Capitolo 4: Comprensione e gestione dei processi, procedure di avvio e spegnimento, differimento di operazioni, operazioni ripetute.
- Capitolo 5: Cosa sono gli standard input, output ed error e come tali caratteristiche vengono utilizzate dalla linea di comando.
- Capitolo 6: Perché dovreste imparare a lavorare con un editor, discussione sugli editor più comuni.
- Capitolo 7: Configurazione del vostro ambiente, grafico, testuale ed audio, impostazioni per gli utenti Linux non di lingua inglese, suggerimenti per aggiungere software extra.
- Capitolo 8: Conversione dei file in formato stampabile, loro stampa, suggerimenti per risolvere problemi di stampa.
- Capitolo 9: Preparazione dei dati per il backup, discussione su vari strumenti, backup remoto.
- Capitolo 10: Panoramica sugli strumenti di rete Linux e applicazioni per gli utenti, con breve disquisizione sui programmi daemon dei servizi di base e connessioni di rete sicure.
- Capitolo 11: In questo capitolo vengono trattati il suono ed il video, compresi Voice over IP e registrazione dei suoni.
- Appendice A: Quali libri da leggere e siti da visitare una volta terminata la lettura di questo testo.
- Appendice B: Un confronto.
- Appendice C: Semmai doveste rimanere bloccati, queste tabelle potrebbero essere una soluzione. Anche un buon argomento quando il vostro capo insiste che VOI dovreste usare la SUA shell favorita.
- Appendice D: Cosa potete fare con questa guida dal punto di vista legale.

Capitolo 1. Cos'è Linux?

Cominceremo con una panoramica su come Linux è diventato l'attuale sistema operativo. Tratteremo degli sviluppi passati e futuri e osserveremo più attentamente i vantaggi e gli svantaggi di questo sistema. Parleremo delle distribuzioni, dell'Open Source in generale e proveremo a spiegare qualcosa su GNU.

Questo capitolo risponde a domande come:

- Cos'è Linux?
- Dove e come è iniziato Linux?
- Linux è un sistema dove tutto si fa mediante linea di comando?
- Linux ha un futuro o è solo una montatura esagerata?
- Quali vantaggi offre Linux?
- Quali gli svantaggi?
- Quali tipi di Linux ci sono e come faccio a scegliere quello adatto a me?
- Cosa sono i movimenti Open Source e GNU?

1.1. Storia

1.1.1. UNIX

Per comprendere la popolarità di Linux dobbiamo tornare indietro nel tempo a circa 30 anni fa...

Immaginate computer grandi come case, oppure come stadi. Mentre le dimensioni di quei computer ponevano problemi sostanziali, c'era una cosa ancora peggiore: ogni computer aveva un differente sistema operativo. Il software veniva sempre adattato per svolgere uno compito determinato e il software di un dato sistema non girava su un altro. Essere capaci di lavorare su di un sistema non significava automaticamente di poter lavorare con un altro. Erano difficoltà sia per gli utenti che per gli amministratori di sistema.

Inoltre, i computer erano estremamente costosi e bisognava compiere sacrifici dopo l'iniziale acquisto anche per spiegare agli utenti come funzionavano. Il costo totale per unità di potenza elaborativa era enorme.

Tecnologicamente il mondo non era abbastanza avanzato, cosicché si continuò con quelle dimensioni per un'altra decade. Nel 1969 un team di sviluppatori dei laboratori Bell Labs cominciò a lavorare su una soluzione per il problema del software, dedicandosi alla questione della compatibilità. Essi svilupparono un nuovo sistema operativo che era:

1. semplice ed elegante;
2. scritto nel linguaggio di programmazione C al posto del codice assembly;

3. capace di riutilizzare il codice.

Gli sviluppatori dei Bell Labs chiamarono il loro progetto “UNIX”.

La caratteristica di poter riutilizzare il codice fu molto importante. Fino ad allora tutti i sistemi di computer commercialmente disponibili erano scritti in un codice specificamente sviluppato per ognuno, UNIX dall'altro lato necessitava solo di un piccolo pezzo di quel codice, ora comunemente chiamato kernel. Tale kernel è l'unico pezzo di codice che bisogna adattare ad ogni specifico sistema e costituisce la base del sistema UNIX. Il sistema operativo e tutte le altre funzioni erano costruite intorno a questo kernel e scritte in un linguaggio di programmazione di più alto livello, il C.

Tale linguaggio fu in particolare sviluppato per creare il sistema UNIX: utilizzando questa nuova tecnica fu molto più semplice sviluppare un sistema operativo che potesse girare su molti tipi diversi di macchine.

I venditori di software furono rapidi ad adattarsi, dal momento che potevano vendere dieci volte di più software quasi senza sforzo. Vennero in essere nuove strane situazioni: immaginate per un momento computer di diversi costruttori comunicanti sulla stessa rete oppure utenti che lavorano su sistemi diversi senza necessità di nuovi studi per usare altri computer. UNIX ha fatto molto per aiutare gli utenti ad essere pronti per computer diversi.

Nella successiva coppia di decadi lo sviluppo di UNIX è proseguito. Parecchie cose sono divenute possibili da fare e parecchi venditori di hardware e software hanno aggiunto il supporto UNIX ai loro prodotti.

UNIX inizialmente si trovava solo in ambienti molto vasti con mainframe e minicomputer (notate che un PC è un “micro” computer). Dovevate lavorare in un'università, per il governo o per società molto facoltose per mettere le mani su un sistema UNIX.

Ma furono sviluppati computer più piccoli ed alla fine degli anni '80 molte persone avevano degli home computer. A quel tempo esistevano diverse versioni di UNIX disponibili per l'architettura PC ma nessuna di loro era realmente libera e, ancora più importante, erano tutte terribilmente lente, cosicché la maggioranza della gente faceva girare MS DOS o Windows 3.11 nei propri computer domestici.

1.1.2. Linus e Linux

Agli inizi degli anni '90 i PC domestici erano finalmente abbastanza potenti da far girare uno UNIX completo. Linus Torvalds, un giovane studente di informatica all'università di Helsinki, ritenne che sarebbe stata una buona idea avere un qualche tipo di versione accademica di UNIX liberamente disponibile, ed immediatamente incominciò a programmare.

Egli cominciò a porre delle domande per cercare risposte e soluzioni che lo potessero aiutare ad avere UNIX sul suo PC. Qui sotto c'è uno dei suo primi messaggi su comp.os.minix, datato 1991:

```
From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: Gcc-1.40 and posix-question
Message-ID: <1991Jul3.100050.9886@klaava.Helsinki.FI>
Date: 3 Jul 91  10:00:50 GMT
Hello netlanders,
Due to a project I'm working on (in minix), I'm interested in the posix standard
definition. Could somebody please point me to a (preferably) machine-readable
format of the latest posix rules? FTP-sites would be nice.
```

Sin dall'inizio lo scopo di Linus fu avere un sistema libero che fosse completamente aderente all'originale UNIX. Questo è il motivo per cui chiedeva gli standard POSIX, essendo POSIX lo standard per UNIX.

In quei giorni non era stato ancora inventato il plug-and-play, ma così tante persone erano interessate ad avere un sistema UNIX che questo non fu un grosso ostacolo. Nuovi driver furono resi disponibili per tutti i tipi di nuovo hardware ad una velocità sempre maggiore. Non appena un nuovo componente hardware era a disposizione, qualcuno lo comprava e lo sottoponeva al Linux test, come progressivamente si andava chiamando il sistema, rilasciando più codice libero per una gamma sempre più ampia di hardware. Questi programmatori non si limitarono ai loro PC: ogni pezzo di hardware che potevano trovare era utile per Linux.

Quelle persone furono chiamate "nerd" o "freak", ma a loro non interessava altro che la lista dell'hardware supportato crescesse sempre più. Grazie a quella gente Linux ora è non solo ideale da far girare sui nuovi PC, ma anche il sistema preferito per hardware vecchio ed "esotico" che sarebbe stato inutile senza l'esistenza di Linux.

Due anni dopo il messaggio di Linus c'erano già 12000 utenti Linux. Il progetto, popolare tra gli appassionati, crebbe in fretta, rimanendo per tutto il tempo legato agli standard POSIX. Tutte le caratteristiche di UNIX furono aggiunte nei successivi due anni, divenendo così il maturo sistema Linux odierno. Linux è un clone totale di UNIX, disegnato per l'uso su workstation, così come per i server di medio e alto livello. Oggi molti dei principali operatori nel mercato hardware e software hanno il proprio team di sviluppatori Linux; presso i vostri venditori locali voi potete anche acquistare sistemi con Linux preinstallato insieme al supporto ufficiale – sebbene ci sia ancora una quantità di hardware e software che non è supportato.

1.1.3. Attuale utilizzazione dei sistemi Linux

Oggi Linux ha raggiunto il mercato desktop. Gli sviluppatori Linux si sono concentrati sulle reti e sui servizi di base cosicché le applicazioni d'ufficio sono state l'ultima barriera abbattuta. Non ci piace ammettere che Microsoft stia guidando il mercato e perciò un mucchio di alternative sono sorte negli ultimi due anni per rendere Linux una scelta accettabile come workstation, fornendo una comoda interfaccia utente ed applicazioni per ufficio compatibili MS come elaboratori testi, fogli elettronici, presentazioni e simili.

Sul lato server Linux è ben noto come piattaforma stabile ed affidabile, che fornisce database e servizi di commercio elettronico a compagnie come Amazon, il celebre negozio di libri online, le Poste statunitensi, l'Esercito tedesco ed altri ancora. In particolar modo i provider internet e di servizi internet si sono appassionati a Linux impiegato come firewall, proxy e web server, e

troverete una Linux box a disposizione di ogni amministratore di sistema UNIX che apprezzi una confortevole stazione di gestione. Cluster (ndt. gruppi/complessi) di macchine Linux sono state utilizzate nella creazione di film come “Titanic”, “Shrek” ed altri. Negli uffici postali (sono i centri nevralgici che instradano la posta e nel grande motore di ricerca) si usano i cluster per le ricerche internet. Questi sono solo alcuni delle migliaia di compiti svolti giornalmente da Linux in tutto il mondo.

E' anche degno di nota che il moderno Linux non solo gira nelle workstation, nei server medio-grandi, ma anche su aggeggi come palmari, portatili, vagonate di applicazioni incorporate e pure su orologi da polso sperimentali. Ciò fa di Linux l'unico sistema operativo al mondo in grado di ricoprire una così ampia gamma di hardware.

1.2. L'interfaccia utente

1.2.1. Linux è difficile?

Se Linux è difficile da apprendere, dipende dalla persona a cui vi rivolgete: utenti esperti di UNIX vi diranno di no, poiché Linux è un sistema operativo ideale per utenti professionali e programmatori dal momento che è stato e viene sviluppato proprio da queste persone.

E' disponibile tutto ciò che possa desiderare un buon programmatore: compilatori, librerie, strumenti di sviluppo e correzione. Questi pacchetti si trovano in qualsiasi distribuzione Linux. Il compilatore C è compreso gratuitamente – diversamente da molte distribuzioni UNIX che domandano dei compensi per l'uso di questo strumento. Ci sono tutti i documenti e i manuali e spesso sono inclusi esempi per aiutarvi ad iniziare senza sprechi di tempo. Sembra UNIX ed il passaggio da UNIX a Linux è una cosa naturale.

Ai primordi di Linux essere un esperto era un requisito per poter utilizzare il sistema. Quelli che dominavano Linux si sentivamo migliori del resto dei “luser” che non avevano ancora visto la luce. Era pratica comune rispondere “RTFM” (“leggete i [fottuti] manuali”) ai principianti. Mentre i manuali si trovavano in ogni sistema, era molto difficile trovare la documentazione e sebbene qualcuno lo facesse, le spiegazioni erano in termini così tecnici che il nuovo utente facilmente si scoraggiava dal conoscere il sistema.

La comunità di utenti Linux cominciò a comprendere che, se Linux doveva diventare un importante giocatore nel mercato dei sistemi operativi, avrebbero dovuto esserci alcuni grossi cambiamenti nell'accessibilità del sistema.

1.2.2. Linux per utenti non esperti

Società come RedHat, SuSE e Mandriva sono improvvisamente apparse fornendo confezioni di distribuzioni Linux destinate al consumo di massa. Esse hanno integrato una grande quantità di interfacce grafiche per utenti [ndt. GUI = Graphical User Interface] sviluppate dalla comunità per

facilitare la gestione di programmi e servizi. Attualmente come utenti Linux avete tutti mezzi per conoscere a fondo come funziona il vostro sistema, ma non è più necessario possedere tale conoscenza per adattare il sistema alle vostre esigenze.

Oggigiorno potete registrarvi in modalità grafica ed avviare tutte le applicazioni richieste senza necessità di battere un solo carattere, pur tuttavia mantenendo la facoltà di accedere quando necessario al cuore del sistema. Proprio per la sua struttura, Linux consente all'utente di interagire con il sistema: esso si adatta sia agli utenti esperti che a quelli principianti. I nuovi utenti non sono costretti a fare cose complicate, mentre gli utenti veterani non devono sforzarsi di lavorare come quando hanno iniziato a conoscere Linux.

Mentre continua lo sviluppo nel settore dell'assistenza, grandi passi sono stati compiuti per gli utenti desktop, generalmente considerati come la categoria meno interessata ad apprendere il funzionamento di un sistema operativo. Sviluppatori di applicazioni desktop stanno facendo sforzi incredibili per produrre i più bei desktop che voi abbiate mai visto o per far assomigliare le vostre macchine Linux alle vostre precedenti workstation MS Windows o Apple. Gli ultimi sviluppi includono anche il supporto per l'accelerazione 3D e per le periferiche USB, aggiornamenti con un solo clic e pacchetti, ecc... Linux ha tutto ciò e tenta di presentare tutte le sue funzionalità disponibili in una forma logica che sia comprensibile anche per la gente comune. Qui sotto c'è un breve elenco di alcuni esempi importanti. Questi siti presentano molte schermate che vi daranno un'idea di quale Linux sia preferibile nel computer da scrivania:

- http://www.gnome.org
- http://kde.org/screenshots/
- http://www.openoffice.org
- http://www.mozilla.org

1.3. Linux ha futuro?

1.3.1. Open Source

L'idea di fondo del c.d. Software Open Source [ndt.= a codice sorgente aperto] è piuttosto semplice: quando i programmatori possono leggere, distribuire e modificare un programma, questo arriverà ad essere maturo. La gente può adattarlo, individuarne e correggerne gli errori, e ciò ad una velocità tale da ridicolizzare il rendimento degli sviluppatori di programmi appartenenti a società convenzionali. Tale programma sarà più flessibile e di qualità superiore rispetto a quello sviluppato secondo i canali tradizionali perché molte più persone lo avranno testato nelle condizioni più disparate rispetto a ciò che può mai fare lo sviluppatore di software chiuso.

L'iniziativa dell'Open Source ha incominciato a far capire ciò al mondo commerciale e, piuttosto lentamente, i produttori commerciali stanno iniziando a considerare la questione. Mentre molti accademici e tecnici si sono convinti già da vent'anni che questa è la strada da percorrere, i venditori commerciali hanno avuto bisogno di applicazioni come internet per capire la possibilità di realizzare profitti con l'Open Source. Ora Linux ha superato la fase in cui era quasi esclusivamente un sistema accademico, utile soltanto ad una manciata di soggetti con preparazione tecnica.

Attualmente Linux fornisce qualcosa di più di un sistema operativo: esiste un'intera infrastruttura a sostegno della catena di sforzi di creazione del sistema operativo, di realizzazione e prova dei programmi per esso, di distribuzione del tutto agli utenti, di fornitura di assistenza, aggiornamenti, supporto e personalizzazione, ecc... Ora Linux è pronto per la sfida in un mondo che cambia rapidamente.

1.3.2. Dieci anni di esperienza al vostro servizio

Mentre Linux è probabilmente l'iniziativa Open Source più nota, esiste un altro progetto che ha contribuito enormemente alla popolarità di questo sistema operativo: si tratta del progetto chiamato SAMBA e il suo successo è la ricostruzione del c.d. Protocollo Server Message Block (SMB)/Common Internet File System (CIFS), impiegato per fornire file e stampe nelle macchine di classe PC, nativamente supportato da MS Windows NT, OS/2 e Linux. I pacchetti sono ora a disposizione di quasi ogni sistema e forniscono soluzioni di interconnessione in ambienti misti utilizzanti protocolli MS Windows e server di file e di stampa compatibili Windows (compreso WinXP).

Forse più famoso di quello SAMBA è il progetto di server HTTP Apache. Il server gira sotto UNIX, Windows NT e molti altri sistemi operativi. Originariamente conosciuto come “A PatCHy server”, basato su un programma esistente e una serie di “patch files”, il nome del codice maturo merita di essere connotato con il nome della tribu americana degli Apache, celebre per la sua superiore capacità nella strategia bellica e per l'inesauribile resistenza. E' stato dimostrato che Apache è sostanzialmente più veloce, più robusto e più ricco di funzionalità rispetto a molti altri server di rete. Apache gira su siti visitati da milioni di persone al giorno e, mentre nessun supporto ufficiale viene fornito dagli sviluppatori, la comunità degli utenti Apache risponde a tutte le vostre domande: attualmente diverse ditte esterne lo supportano sul piano commerciale.

Nella categoria delle applicazioni da ufficio viene offerta una scelta di cloni della suite MS Office che va da parziali ad integrali sostituti delle applicazioni disponibili nelle stazioni di lavoro [*workstation*] MS Windows. Tali iniziative hanno giovato assai all'accettazione di Linux nel mercato dei desktop perché gli utenti non necessitano di corsi extra per imparare a lavorare con i nuovi sistemi. Attraverso i desktop giungono le preghiere degli utenti comuni, e non soltanto quelle, ma anche le loro richieste particolari, che si fanno sempre più sofisticate e numerose di giorno in giorno.

La comunità Open Source, essendo formata in maggioranza da soggetti che hanno dato il loro contributo per oltre un quinquennio, garantisce la posizione di Linux sia come importante competitore nel mercato dei desktop, sia delle applicazioni generali dell'IT. Dipendenti stipendiati e volontari lavorano diligentemente in modo simile cosicché Linux possa mantenere il suo posto nel mercato. Maggiori utenti, maggiori domande. La comunità Open Source si prodiga per fornire risposte sicure e controlla la qualità delle stesse con occhio attento, con conseguente maggiore stabilità e accessibilità.

Elencare tutto il software Linux disponibile va oltre lo scopo di questa guida dal momento che esistono decine di migliaia di pacchetti. Attraverso questo corso vi presenteremo i pacchetti

software più comuni, che sono per lo più liberamente utilizzabili. Per togliere un po' del panico ai nuovi utenti, ecco una schermata di uno dei programmi più ricercati. Potete constatare da voi stessi che non si sono risparmiati gli sforzi per far sentire come a casa gli utenti che abbandonano Windows:

Figura 1-1. Il foglio elettronico compatibile MS di OpenOffice

Produzione 2005

Mesi	Carta	Ferro	Plastica
gennaio	112	3232	4343
febbraio	43	4332	423
marzo	2223	2265	6544
aprile	565	7666	645
maggio	6	654	654
giugno	6767	456	6555
luglio	76	4645	6554
agosto	76	55654	653
settembre	7666	774	333
ottobre	5456	654	65
novembre	655	64	534
dicembre	65	464	544
Totale	23710	80860	27847

1.4. Caratteristiche di Linux

1.4.1. Vantaggi di Linux

Molti dei vantaggi di Linux dipendono dalle sue origini, profondamente radicate in UNIX, ad eccezione naturalmente del primo vantaggio:

- Linux è gratuito

  Come una birra gratis, dicono. Se non volete spendere assolutamente nulla, non dovete neppure pagare il prezzo di un CD: Linux può essere scaricato interamente da internet in modo gratuito. Non ci sono tasse di registrazione, niente costi per utente, aggiornamenti gratuiti e codice sono liberamente a disposizione se volete cambiare il comportamento del vostro sistema.

  Soprattutto Linux è libero come è libera la parola:

  La licenza comunemente adottata è la GNU Public License (GPL). La licenza

afferma che chiunque voglia farlo, ha il diritto di modificare Linux ed eventualmente di redistribuire una versione modificata, all'unica condizione che il codice sia ancora a disposizione dopo tale redistribuzione,. In pratica siete liberi di prendere una immagine del kernel, ad esempio per aggiungere il supporto per le macchine di teletrasporto o di viaggio nel tempo e di vendere il vostro nuovo codice purché il vostro cliente possa avere una copia di quel codice.

- Linux è portabile su qualsiasi piattaforma hardware

Un produttore che voglia vendere un nuovo tipo di computer e non sa che genere di OS girerà sulla sua nuova macchina (sia che si tratti della CPU della vostra auto o della lavatrice) può prendere un kernel Linux e renderlo funzionante sul suo hardware dal momento che la documentazione relativa a tale attività è liberamente disponibile.

- Linux è stato progettato per rimanere sempre funzionante

Come con UNIX, ci si aspetta che un sistema Linux resti in esecuzione per tutto il tempo senza necessità di riavviarlo: ciò perché una quantità di operazioni vengono svolte di notte o rispettando automaticamente orari prefissati in altri momenti di calma, consentendo così una maggiore disponibilità nei periodi di carico maggiore ed un uso più bilanciato dell'hardware. Questa caratteristica permette di applicare Linux in ambienti dove le persone non hanno il tempo o la possibilità di controllare i propri sistemi giorno e notte.

- Linux è sicuro e versatile

Il modello di sicurezza utilizzato da Linux si basa sul concetto di sicurezza UNIX, conosciuto per la sua robustezza e la sua sperimentata qualità. Ma Linux non si adatta solo ad essere un baluardo contro gli attacchi nemici provenienti da internet: infatti esso si adegua ad altre situazioni utilizzando i medesimi elevati standard di sicurezza. La vostra macchina di sviluppo o la stazione di controllo saranno sicure come il vostro firewall.

- Linux è scalabile

Da un Palmtop da 2 MB di memoria ad un gruppo di archiviazione da un petabyte con centinaia di nodi: aggiungete o togliete i pacchetti appropriati e Linux si adatterà ad entrambi. D'altra parte non avete bisogno di un supercomputer, perché potete usare Linux per grossi compiti utilizzando i “mattoni” forniti con il sistema. Se volete fare cose piccole come la realizzazione di un sistema operativo per un processore integrato o solo riciclare il vecchio 486, Linux ci riuscirà al meglio.

- Il sistema operativo Linux e le sue applicazioni hanno tempi di debug molto brevi

Siccome Linux è stato sviluppato e provato da migliaia di persone, sia gli errori sia le persone per correggerli si trovano normalmente piuttosto in fretta. Qualche volta accade che trascorrano solo due ore dalla scoperta alla correzione di un bug.

1.4.2. Svantaggi di Linux

- Ci sono troppe differenti distribuzioni

"Quot capites, tot rationes", come già dicevano i latini: tante persone, tante opinioni. Di primo acchito la quantità di distribuzioni Linux potrebbe sembrare spaventosa o ridicola a seconda del vostro punto di vista. D'altro canto ciò significa che ognuno trova quella che gli serve. Non avete bisogno di essere degli esperti per trovare una versione adatta.

Quando gli viene richiesto, generalmente ogni utente Linux risponderà che la miglior distribuzione è quella specifica versione che sta usando. Così quale andrebbe scelta? Non preoccupatevi eccessivamente di ciò: tutte le versioni contengono più o meno lo stesso complesso di pacchetti base. Oltre a quest'ultimi, vengono aggiunti software di terze parti rendendo così, ad esempio, TurboLinux più adatta alla piccola e media impresa, RedHat per i server e SuseLinux per le workstation. Comunque le differenze sono, il più delle volte, solo superficiali. La miglior strategia è quella di provare una coppia di distribuzioni: peccato che non tutti abbiano il tempo per questa prova. Fortunatamente esiste una marea di consigli sull'argomento della scelta del vostro Linux. Una rapida ricerca su Google utilizzando le parole "choosing your distribution" [ndt. "scelta della distribuzione] vi darà oltre una decina di collegamenti a buoni suggerimenti. L'Installation HOWTO tratta anche della scelta della distribuzione.

- Linux non è molto facile e confonde i nuovi utenti

Va detto che Linux, perlomeno il nucleo del sistema, è meno semplice rispetto a MS Windows e sicuramente molto più complicato di MacOS, ma... alla luce della sua popolarità sono stati compiuti sforzi considerevoli per rendere Linux sempre più facile da usare, specialmente per i novizi. Sempre più informazioni vengono rilasciate quotidianamente, come questa guida, per aiutare a colmare la disparità di documentazione disponibile per gli utenti di tutti i livelli.

- Un prodotto Open Source è affidabile?

Come può una cosa che è gratuita essere anche affidabile? Gli utenti Linux possono scegliere di usarlo o fare a meno, cosa che da loro un grosso vantaggio rispetto agli utenti di software proprietario, che non posseggono questo tipo di libertà. Dopo lunghi periodi di prova, molti utenti Linux giungono alla conclusione che Linux non solo è valido ma in molti casi migliore e più veloce delle soluzioni tradizionali. Se Linux non fosse stato affidabile, sarebbe scomparso molto tempo fa, senza conoscere la popolarità odierna con milioni di utenti. Ora gli utenti possono modificare i propri sistemi e condividere le proprie osservazioni con la comunità, cosicché il sistema migliora di giorno in giorno. E' un progetto non completato, questo è vero, ma in un mondo in continua evoluzione Linux è un progetto che continua a competere per la perfezione.

1.5. Sapore di Linux

1.5.1. Linux e GNU

Sebbene esista un gran numero di versioni di Linux, troverete molte similitudini tra le differenti distribuzioni, poiché ogni macchina Linux è come un mucchio di mattoni che potete posizionare assieme seguendo le vostre esigenze e gusti. Installare il sistema è solo l'inizio di una relazione a lungo termine: non appena pensate di avere in bel sistema funzionante, Linux stimolerà la vostra immaginazione e creatività e più capirete la potenza che vi viene offerta dal sistema e più cercherete di ridefinire i suoi limiti.

Linux può apparire diverso a seconda della distribuzione, dell'hardware e del gusto personale ma le basi su cui si fondano tutte le interfacce, grafiche o meno, rimangono le medesime. Il sistema Linux è basato sugli strumenti GNU (Gnu's Not Unix), che forniscono un insieme di metodi comuni per maneggiarlo ed usarlo. Tutti gli strumenti GNU [ndt.: GNU tools] sono a sorgente aperto cosicché possono essere installati su qualsiasi sistema. Molte distribuzioni offrono pacchetti precompilati di strumenti molto comuni, come i pacchetti RPM di RedHat o quelli Debian (chiamati anche deb o dpkg) di Debian, in maniera che non dovete essere dei programmatori per installare un pacchetto nel vostro sistema. Comunque, se vi piace fare da voi stessi, godrete di Linux al meglio dal momento che molte distribuzioni sono fornite di un insieme completo di strumenti di sviluppo, che consentono di installare nuovo software semplicemente dal codice sorgente. Tale modalità permette di installare programmi anche se non esistono già preparati per il vostro sistema.

Un elenco di programmi GNU comuni:

- Bash: la shell GNU
- GCC: il Compilatore C di GNU
- GDB: il Debugger GNU
- Coreutils: un insieme di utility base in stile UNIX, come **ls**, **cat** e **chmod**
- Findutils: per cercare e trovare file
- Fontutils: per convertire font da un formato ad un altro e per crearne di nuovi
- The Gimp: GNU Image Manipulation Program [Programma GNU di Manipolazione delle Immagini]
- Gnome: l'ambiente desktop GNU
- Emacs: un editor molto potente
- Ghostscript e Ghostview: interprete e frontend grafico per i file PostScript.
- GNU Photo: programma per interagire con le macchine fotografiche digitali.
- Octave: un linguaggio di programmazione destinato in primo luogo a svolgere calcoli numerici ed elaborazioni di immagini.
- GNU SQL: sistema di database relazionale
- Radius: server remoto di autenticazione e gestione account.
- ...

Molte applicazioni commerciali sono disponibili per Linux (per maggiori informazioni su tali pacchetti ci riportiamo alla loro specifica documentazione). Attraverso questa guida noi tratteremo

solo di programmi liberamente disponibili, forniti (per la maggior parte) di licenza GNU.
Per installare pacchetti cancellati o nuovi avrete bisogno di una qualche forma di gestione del software: le più comuni sono RPM e dpkg. RPM è il RedHat Package Manager [gestore di pacchetti RedHat] utilizzato da svariati sistemi Linux nonostante il nome non lo lasci pensare. Dpkg è il sistema di gestione pacchetti di Debian che fa uso di un'interfaccia chiamata **apt-get** in grado di gestire anche i pacchetti RPM. Novell Ximian Red Carpet è una implementazione di RPM con interfaccia grafica proveniente da una terza parte. Altri produttori di software possono avere le proprie procedure d'installazione, qualche volta somiglianti a InstallShield od altri programmi simili, noti in MS Windows ed altre piattaforme. Quando sarete più esperti di Linux avrete a che fare facilmente con uno o più di questi programmi.

1.5.2. GNU/Linux

Il kernel Linux (le *ossa* del vostro sistema, v. Sezione 3.2.3.1) non fa parte del progetto GNU ma adotta la stessa licenza del software GNU. La grande maggioranza di programmi di utilità e di strumenti di sviluppo (la *carne* del vostro sistema), che non è specifica Linux, è tratta dal progetto GNU. Poiché qualsiasi sistema usabile deve avere sia il kernel che un insieme, anche minimo, di utility, alcune persone affermano che questo sistema dovrebbe essere definito sistema *GNU/Linux*.

Per ottenere il maggior grado possibile di indipendenza tra distribuzioni, tratteremo questo genere di Linux in questo corso. Quando non parleremo di un sistema GNU/Linux, provvederemo ad indicare la specifica distribuzione, la versione o il nome del programma.

1.5.3. Quale distribuzione dovrei installare?

Prima dell'installazione, il fattore più importante è il vostro hardware. Dal momento che ogni distribuzione Linux contiene i pacchetti base e può essere realizzata per adattarsi a quasi tutte le richieste (in quanto esse usano tutte il kernel Linux), avete solo bisogno di capire se la distribuzione girerà sulla vostra macchina. LinuxPPC, per esempio, è stato concepito per funzionare su Apple ed altri PowerPC e non gira su normali PC basati su x86. LinuxPPC funziona sui nuovi Mac ma non è utilizzabile su altri più vecchi con sorpassata tecnologia del bus. Altro caso complicato è l'hardware Sun, che potrebbe essere una vecchia CPU SPARC oppure una più recente UltraSparc: queste richiedono differenti versioni di Linux.

Alcune distribuzioni di Linux sono ottimizzate per certi processori, come la CPU Athlon, mentre contemporaneamente funzioneranno in modo decente su processori Intel486, 586 e 686 standard. Talvolta le distribuzioni per certe CPU speciali non sono affidabili perché provate da poche persone.

Molte distribuzioni Linux offrono un insieme di programmi per PC generici con pacchetti speciali contenenti kernel ottimizzati per le CPU basate su Intel x86. Tali distribuzioni sono ben testate e mantenute regolarmente, focalizzate su una affidabile realizzazione per server e su semplici procedure di installazione ed aggiornamento. Esempi sono Debian, Ubuntu, Fedora, SuSE e

Mandriva che sono di gran lunga i più popolari sistemi Linux e sono in genere considerati semplici da gestire per l'utente principiante, mentre non impediscono ai professionisti di ottenere il massimo dai loro computer Linux. Linux gira decentemente anche sui portatili e sui server di medio livello. I driver per il nuovo hardware vengono inclusi solo dopo prove approfondite, cosa che accresce la stabilità di un sistema.

Mentre il desktop standard su un sistema potrebbe essere Gnome, un altro potrebbe offrire KDE di base. Generalmente sia Gnome e KDE sono disponibili in tutte le principali distribuzioni Linux. Altri gestori di finestre e desktop sono a disposizione degli utenti più avanzati.

Il processo di installazione standard consente agli utenti di scegliere tra differenti impostazioni base, come quella workstation in cui tutti i pacchetti necessari all'uso quotidiano e per lo sviluppo, o come quella server in cui molteplici servizi di rete possono essere selezionati. Gli utenti esperti possono installare qualsiasi combinazione di pacchetti che desiderano durante il processo iniziale di installazione.

L'obbiettivo di questa guida è di applicarsi a tutte le distribuzioni Linux. Per vostra comodità, comunque, si consiglia vivamente che i principianti si affidino ad una delle distribuzioni principali, che supportano tutto l'hardware comune e le applicazioni di base. Quelle seguenti sono ottime scelte per i novizi:

- Fedora Core
- Debian
- SuSE Linux
- Mandriva (in precedenza MandrakeSoft)
- Knoppix: un sistema operativo che funziona dal vostro CD-ROM senza la necessità di installare alcunché.

Immagini ISO scaricabili si possono trovare su LinuxISO.org. Le principali distribuzioni possono essere acquistate presso qualsiasi decente negozio di computer.

1.6. Sommario

In questo capitolo abbiamo appreso che:

- Linux è una specie di UNIX.
- Il sistema operativo Linux è scritto nel linguaggio di programmazione C.
- “De gustibus et coloribus disputandum non est”: esiste un Linux per ciascuno.
- Linux utilizza gli strumenti GNU, un complesso di strumenti standard liberamente disponibili per la manipolazione del sistema operativo.

1.7. Esercizi

Un esercizio pratico per chi inizia: installate Linux nel vostro PC. Leggete il manuale di installazione della vostra distribuzione e/o l'Installation HOWTO e procedete.

> **Leggete la documentazione!**
> Molti errori vi bloccano per non aver letto le informazioni fornite durante l'installazione. Leggere perciò con attenzione i messaggi di installazione è il primo passo sulla strada del successo.

Cose da sapere PRIMA di iniziare un'installazione di Linux:

- Questa distribuzione girerà sul vostro hardware?
  Controllate su http://www.tldp.org/HOWTO/Hardware-HOWTO/index.html in caso di dubbi circa la compatibilità del vostro sistema.
- Che genere di tastiera possiedo (numero di tasti, formato)? Che tipo di mouse (seriale/parallelo, numero di pulsanti)? Quanti MB di RAM?
- Installerò una workstation base o un server, oppure dovrò selezionare da me stesso pacchetti specifici?
- Installerò dal mio disco rigido, da un CD-ROM o attraverso la rete? Dovrò adeguare il BIOS per ciascuno di questi? Il metodo di installazione richiede un disco di avvio?
- Linux sarà l'unico sistema operativo sul computer oppure si procederà ad un'installazione per l'avvio di due sistemi alternativamente? Dovrò creare una grande partizione per installare più avanti dei sistemi virtuali o questa è essa stessa un'installazione virtuale?
- Il computer è in rete? Qual è il suo nome di host, il suo indirizzo IP? C'è qualche server gateway o altra importante macchina in rete con cui comunicare?

  > **Linux si attende di essere collegato in rete**
  > Non utilizzare la rete o configurarla in modo sbagliato può causare un avvio lento.
- Il computer è un gateway/router/firewall? (Se vi ponete questa domanda, probabilmente non lo è).
- Partizionamento: per questa volta lasciate svolgere il compito al programma di installazione: tratteremo le partizioni in dettaglio nel Capitolo 3. Esiste documentazione specifica del sistema da installare se volete sapere tutto di esso. Se la vostra distribuzione Linux non consente il partizionamento automatico, significa probabilmente che non è adatta ai principianti.
- Questa macchina si avvierà in modalità testo o grafica?
- Pensate ad una buona parola-chiave per l'amministratore di questa macchina (root). Create un account di utente non-root (accesso non privilegiato al sistema).
- Ho bisogno di un disco di ripristino [rescue disk]? (operazione raccomandata)
- Che lingua voglio?

Tutta la lista di controllo si può trovare su http://www.tldp.org/HOWTO/Installation-HOWTO/index.html.

Nei successivi capitoli scopriremo se l'installazione è andata a buon fine.

Capitolo 2. Avvio rapido

Per trarre il massimo da questa guida, cominceremo subito con un capitolo pratico sulla connessione ad un sistema Linux compiendo alcune operazioni fondamentali.

Tratteremo di:

- Connessione al sistema
- Disconnessione dal sistema
- Modalità testo e grafica
- Cambio della password
- Navigazione nel file system
- Determinazione del tipo di file
- Osservazioni nei file di testo
- Ricerca di aiuti

2.1. Connettersi, attivare l'interfaccia utente e disconnettersi

2.1.1. Introduzione

Per lavora direttamente in un sistema Linux avrete bisogno di un nome utente e di una password. Dovrete sempre autenticarvi nel sistema. Come già ricordato nell'esercizio del Capitolo 1, molti sistemi Linux basati su PC hanno due modalità fondamentali per avviarsi: o in rapida e sobria modalità console testuale, che sembra come il DOS con mouse, dotata di caratteristiche multitasking e multiutente, o in quella grafica che ha un aspetto migliore ma sottrae molte più risorse di sistema.

2.1.2. Modalità grafica

Oggigiorno è la modalità standard sulla maggior parte dei computer desktop. Voi sapete che vi connetterete al sistema utilizzando la modalità grafica dopo che vi sarà stato richiesto per prima cosa il nome utente e poi la password.

Per autenticarvi, assicuratevi che il puntatore del mouse sia nella finestra di login, fornite il vostro nome utente e password al sistema e premete OK o il tasto **Invio**.

Attenti con quell'account di root!

Generalmente è considerata una pessima idea connettersi (in grafica) utilizzando il nome utente *root*, l'account dell'amministratore del sistema, dal momento che l'uso della grafica comprende l'avvio di una quantità extra di programmi, in caso di utente *root* con

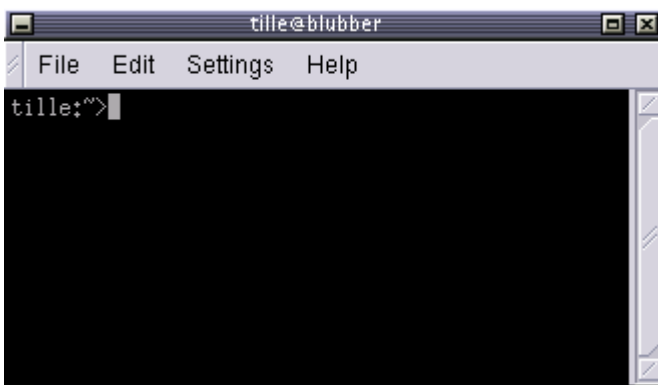
> un sacco di permessi extra. Per mantenere il rischio più basso possibile, usate un account di utente normale per connettervi graficamente. Ci sono così tanti rischi nel connettersi con account *root* che conviene tenere in mente ciò come regola generale: connettetevi come *root* solo quando servono privilegi extra.

Dopo aver inserito la combinazione nome/password, è necessario attendere un attimo prima dell'avvio dell'ambiente grafico, in base alla velocità della CPU del vostro computer, al software che utilizzate ed alle impostazioni personali.

Per continuare dovrete aprire una *terminal window* [finestra di terminale] o *xterm* in breve (X è il nome del software che supporta il sottostante ambiente grafico). Tale programma si può trovare sotto Applicazioni->Utilità File, Strumenti di Sistema o il menu Internet, a seconda del gestore di finestra impiegato. Potrebbero esserci delle icone da usare come scorciatoia per ottenere una finestra xterm, così come premendo il tasto destro del mouse sullo sfondo normalmente apparirà un menu contenente una applicazione di finestra di terminale.

Scorrendo i menu noterete che si possono fare molte cose senza introdurre comandi via tastiera. Per molti utenti andrà bene il buon vecchio metodo di gestire il computer "punta_e_clicca". Ma questa guida è destinata ai futuri amministratori di rete e di sistema, i quali dovranno occuparsi del cuore del sistema. Essi necessitano di uno strumento più avanzato rispetto al mouse per gestire tutti i compiti che dovranno affrontare. Tale strumento è la shell e, trovandoci in modalità grafica, la attiveremo aprendo una finestra di terminale.

La finestra di terminale è il vostro pannello di controllo del sistema. Quasi tutte le seguenti operazioni saranno svolte utilizzando questo semplice ma potente strumento testuale. Una finestra di terminale dovrebbe sempre mostrare il prompt dei comandi alla sua apertura. Il seguente terminale mostra un prompt standard, che fa apparire il nome di login dell'utente e l'attuale directory di lavoro, rappresentata dalla tilde (~):

Figura 2-1. Finestra di terminale

Un'altra forma comune per un prompt è questa:

```
[utente@host dir]
```

Nell'esempio soprastante, *utente* è il vostro nome di login, *host* il nome del computer su cui state

lavorando e *dir* un'indicazione dell'attuale locazione nel file system.

Più avanti tratteremo dettagliatamente dei prompt e del loro comportamento: per adesso è sufficiente sapere che essi posso mostrare ogni tipo di informazione ma non fanno parte dei comandi che date al sistema.

Per disconnettervi dal sistema in modalità grafica dovete chiudere tutte le finestre di terminale e le altre applicazioni. Dopo di ciò premete l'icona di logout o cercate Log Out nel menu. In realtà chiudere tutto non è necessario e il sistema può farlo al posto vostro, ma il gestore di sessione potrebbe riportare tutte le applicazioni attualmente aperte di nuovo a video nel successivo collegamento, rallentando il sistema, effetto questo non sempre desiderato. Comunque tale comportamento è configurabile.

Quando vedrete di nuovo la schermata di login richiedere nome utente e password, capirete che la disconnessione ha avuto successo.

Gnome o KDE?
Abbiamo citato già un paio di volte i desktop Gnome e KDE. Si tratta dei due modi più diffusi per la gestione del vostro desktop [scrivania elettronica], sebbene ne esistano altri, molti altri. Qualsiasi desktop scegliate di utilizzare, va bene (se sapete come aprire una finestra di terminale). Comunque continueremo a fare riferimento sia a Gnome che a KDE quali modi più popolari per svolgere determinati compiti.

2.1.3. Modalità testo

Capirete di essere in modalità testo quando l'intero schermo sarà nero con caratteri (per lo più bianchi). Uno schermo di login in modo testuale mostra tipicamente alcune informazioni sulla macchina con cui state lavorando, il nome di questa e un prompt che vi attende per l'autenticazione:

```
RedHat Linux Release 8. (Psyche)

blast login: _
```

Il login è diverso da quello grafico in quanto dovete battere il tasto **Invio** dopo aver fornito il vostro nome utente poiché non ci sono pulsanti sullo schermo da schiacciare con il mouse. Subito dopo dovete scrivere la vostra password seguita da un altro **Invio**: non vedrete alcuna indicazione di ciò che state digitando, neppure un asterisco, e non vedrete il cursore muoversi. Questo comportamento è normale sotto Linux ed è stato fatto per ragioni di sicurezza.

Una volta che il sistema vi ha accettato come utente valido, potrete eventualmente avere altre informazioni, chiamate messaggio del giorno (*message of the day*). Inoltre è frequente nei sistemi UNIX mostrare un “biscotto della fortuna” (*fortune cookie*) contenente alcuni pensieri saggi o strampalati (ciò spetta a voi). Dopo di questo, avrete a disposizione una shell, indicata con il medesimo prompt che otterreste in modalità grafica.

Non autenticatevi come root

Anche in modalità testo: autenticatevi come root solo per operazioni di setup e configurazione che richiedano assolutamente i privilegi di amministratore, come l'aggiunta di utenti, l'installazione di pacchetti di programmi, il funzionamento della rete ed altre configurazioni di sistema. Una volta terminato, abbandonate immediatamente lo speciale account e riprendete il vostro lavoro come utente non privilegiato. In alternativa, alcuni sistemi, come Ubuntu, vi costringono ad utilizzare **sudo**, cosicché non avete necessità di accedere direttamente all'account amministrativo.

La disconnessione si esegue inserendo il comando **logout** seguito da **Invio**: sarete disconnessi con successo dal sistema quando vedrete nuovamente la schermata di login.

Il pulsante di spegnimento
Dal momento che non ha senso spegnere Linux senza applicare le corrette procedure di spegnimento del sistema, la pressione del tasto di accensione equivale ad avviare quelle procedure su sistemi più nuovi. Comunque, spegnere un vecchio sistema senza eseguire il processo di chiusura può causare gravi danni! Se volete essere al sicuro, utilizzate sempre l'opzione di spegnimento quando chiudete dall'interfaccia grafica oppure, quando siete alla schermata di autenticazione (in cui dovete indicare il vostro nome utente e la parola-chiave), cercate un bottone di spegnimento.

Ora che sappiamo come connetterci e disconnetterci dal sistema, siamo pronti per i nostri primi comandi.

2.2. Rudimenti essenziali

2.2.1. I comandi

Questi sono quelli immediati che ci servono per iniziare; li tratteremo meglio più avanti.

Tabella 2-1. Comandi di avvio rapido

Comando	**Significato**
ls	Mostra un elenco dei file contenuti nella directory di lavoro attuale, come il comando **dir** del DOS
cd `directory`	cambio directory
passwd	cambio della password dell'utente correntemente
file `nomefile`	mostra il tipo di file di quello chiamato `nomefile`
cat `filetesto`	fa apparire sullo schermo il contenuto di `filetesto`
pwd	mostra la directory di lavoro attuale
exit o **logout**	abbandona la sessione

Comando	Significato
man *comando*	legge le pagine man relative a *comando*
info *comando*	legge le pagine info relative a *comando*
apropos ***stringa***	cerca nel database *whatis* la *stringa*

2.2.2. Annotazioni generali

Scrivete questi comandi dopo il prompt in una finestra di terminale, in modalità grafica o in modalità testo, seguiti da **Invio**.

I comandi possono essere eseguiti da soli, come il comando **ls**. Un comando si comporta differentemente quando si aggiunge una *opzione*, normalmente preceduta da un segno meno (-), come **ls -a**. Il medesimo carattere di opzione può assumere significati diversi con altri comandi. I comandi GNU accettano opzioni estese, precedute da due meno (--), come **ls --all**. Alcuni comandi non hanno opzioni.

L'argomento di un comando è una precisazione circa l'oggetto su cui volete che il comando agisca: un esempio è **ls** ***/etc***, dove la directory */etc* è l'argomento del comando **ls**. Ciò indica che volete vedere il contenuto di quella directory, invece di quella normale ottenibile battendo il semplice comando **ls** seguito da **Invio**. Qualche volta i comandi richiedono degli argomenti, qualche altra gli argomenti sono solo opzionali.

Potete scoprire se un comando accetta opzioni e argomenti, e quali di questi sono validi, controllando gli aiuti in linea per quel comando (v. Sezione 2.3).

In Linux, come in UNIX, le directory sono separate utilizzando delle barre (/) come quelle usate per gli indirizzi di rete (URL): più avanti tratteremo la struttura delle directory approfonditamente.

I simboli **.** e **..** hanno un significato speciale quando riguardano le directory: proveremo a scoprirlo con gli esercizi e ancor più nel prossimo capitolo.

Cercate di evitare di autenticarvi o di usare l'account di amministratore *root*. Oltre a svolgere il vostro normale lavoro, molti compiti, inclusi il controllo del sistema, la raccolta di informazioni, ecc., possono essere eseguiti con un account di utente normale senza alcuna necessità di permessi speciali. Se necessario, per esempio quando create un nuovo utente o installate nuovo software, il modo preferibile per ottenere l'accesso di root è attraverso lo scambio degli ID di utente (v. Sezione 3.2.1 per un esempio).

Quasi tutti i comandi di questo libro possono essere eseguiti senza i privilegi si amministratore di sistema. In molti casi quando date un comando o avviate un programma come utente non privilegiato, il sistema vi avvisa o vi presenta la richiesta della password di root se è necessario l'accesso di root. Una volta fatto, abbandonate immediatamente l'applicazione o la sessione che vi

ha fornito i privilegi di root.

Leggere la documentazione dovrebbe diventare la vostra seconda natura. Specialmente all'inizio è importante leggere la documentazione di sistema, i manuali dei comandi base, gli HOWTO e così via. Dal momento che la quantità di documentazione è così vasta, è impossibile inserire tutta la relativa documentazione. Questo libro proverà a orientarvi verso la documentazione più appropriata su ogni argomento trattato per stimolare anche l'abitudine a leggere le pagine man.

2.2.3. Usare le caratteristiche di Bash

Alcune combinazioni speciali di tasti vi consentono di fare cose più facilmente e più rapidamente con la shell GNU, Bash, la quale si trova presente di norma in quasi tutti i sistemi Linux (v. Sezione 3.2.3.2). Qui sotto c'è una lista delle funzioni più utilizzate: siete fortemente consigliati ad abituarvi al loro uso in modo da ottenere il massimo di esperienza Linux sin dal principio.

Tabella 2-2. Combinazioni di tasti in Bash

Tasto o combinazione di tasti	Funzione
Ctrl+A	Muove il cursore all'inizio della linea di comando.
Ctrl+C	Termina un programma attivo e ritorna al prompt (v. Capitolo 4).
Ctrl+D	Disconnessione dalla corrente sessione di shell: corrisponde alla scrittura di **exit** o **logout**.
Ctrl+E	Sposta il cursore in fondo alla linea di comando.
Ctrl+H	Genera un carattere di backspace [cancellazione all'indietro].
Ctrl+L	Pulisce il terminale.
Ctrl+R	Ricerca nella cronologia [history] dei comandi (v. Sezione 3.3.3.4).
Ctrl+Z	Sospende un programma (v. Capitolo 4)
FrecciaSinistra e **FrecciaDestra**	Sposta il cursore di uno spazio a sinistra o a destra sulla linea di comando in modo che potete inserire caratteri in altri posti oltre a quelli d'inizio e fine.
FrecciaSu e **FrecciaGiù**	Scorre la cronologia [history] dei comandi. Andate alla linea che volete ripetere, modificate i dettagli se necessario e premete Invio per risparmiare tempo.
Maiuscolo+PaginaSu e **Maiuscolo+PaginaGiù**	Scorre il buffer di terminale (per vedere il testo che ha "spostato" lo schermo).
Tab	Completamento di comandi o nomi di file: quando sono possibili più scelte, il sistema ve lo segnalerà con un segnale sonoro o visivo, altrimenti, se le scelte sono troppe, vi chiederà se volete vedere tutte quante.

Tasto o combinazione di tasti	Funzione
Tab Tab	Mostra le possibilità di completamento di file o comandi.

Le ultime due voci nella soprastante tabella richiedono alcune spiegazioni extra. Per esempio, se volete spostarvi nella directory `directory_dal_nome_piuttosto_lungo`, non dovete digitare assolutamente tutto quel lunghissimo nome. Dovete solo battere nella linea di comando **cd dir** e poi premere il tasto **Tab**: la shell provvedere a completare il nome per voi se non esistono altri file che iniziano con gli stessi tre caratteri. Naturalmente se non esistono altre parole che iniziano con “d”, allora potete digitare solamente **cd d** e poi **Tab**. Se più di un file inizia con gli stessi caratteri, la shell ve lo segnalerà, dopo di che potrete battere due volte **Tab** di seguito e la shell mostrerà le scelte disponibili:

```
vostro_prompt> cd st
starthere stuff stuffit
```

Nel esempio qui sopra se digiterete “a” dopo i primi due caratteri e batterete **Tab** nuovamente, non rimarranno altre possibilità e la shell completerà il nome della directory senza costringervi a scrivere la stringa “rthere”:

```
vostro_prompt> cd starthere
```

Naturalmente dovrete premere **Invio** per accettare la scelta.

Nello stesso esempio, se digitate “u” e poi battete **Tab,** la shell aggiungerà “ff” per voi, ma poi protesterà nuovamente perché sono possibili più scelte. Premendo ancora **Tab Tab**, vedrete le scelte; se batterete uno o più caratteri, in modo da rende univoca la scelta al sistema, e **Tab** di nuovo (o **Enter** quando avete raggiunto la fine del nome del file da voi scelto), la shell completerà il nome del file e vi sposterà in quella directory - se naturalmente si tratta di un nome di directory.

Ciò funziona con tutti i nomi dei file che sono argomenti dei comandi.

La stessa cosa succede per il completamento dei nomi dei comandi. Digitando **ls** e battendo due volte il tasto **Tab**, apparirà l'elenco di tutti i comandi del vostro `PATH` (v. Sezione 3.2.1) che iniziano con quelle due lettere:

```
vostro_prompt> ls
ls            lsdev  lspci      lsraid     lsw
lsattr        lsmod  lspgpot               lss16toppm
lsb_release  lsof               lspnp                  lsusb
```

2.3. Cercare aiuto

2.3.1. State attenti

GNU/Linux è tutto teso a diventare più autoesplicativo. E, come è consuetudine con questo sistema, ci sono diversi modi per raggiungere lo scopo. Un modo comune per ricevere aiuto è trovare qualcuno che sappia e, sebbene la comunità di utenti Linux sia paziente ed amante della pace, quasi tutti si aspetteranno comunque che voi abbiate già provato uno o più metodi di questa sezione prima di interpellarli: i modi in cui viene espresso questo punto di vista possono essere piuttosto bruschi se non dimostrate di aver seguito questa regola fondamentale.

2.3.2. Le pagine man

Molti principianti temono le pagine man (manuale), perché esse sono una fonte travolgente di documentazione. Queste però sono ben organizzate, come potrete verificare nell'esempio seguente: **man man**.

La lettura delle c.d. *man pages* si effettua normalmente in una finestra di terminale (in modalità grafica) o semplicemente in modalità testo se lo preferite. Dopo il prompt scrivete questo comando seguito da **Invio**:

```
vostronome@vostrocomp ~> man man
```

La documentazione di **man** apparirà sullo schermo dopo aver premuto **Invio**:

```
man(1)                                                        man(1)



NOME
          man - formatta formatta e mostra le pagine di guida in linea
          manpath  -  determina  i  percorsi  di  ricerca  dell'utente  per  le
pagine di guida

SINTASSI

       man [-acdfFhkKtwW] [-m sistema] [-p stringa]  [-C  file_configurazione]
       [-M percorso] [-P impaginatore] [-S lista_sezioni] [sezione] nome ...

DESCRIZIONE
       man  formatta  e  mostra  le  pagine di guida in linea. Questa versione
       riconosce le variabili d'ambiente MANPATH  e  (MAN)PAGER,  in  modo  da
       avere  il  proprio  insieme di pagine di guida e scegliere il programma
       preferito per leggere le pagine formattate. Se sezione  è  specificata,
       man cerca solamente in quella sezione del manuale.  Si può anche speci-
       ficare l'ordine di ricerca delle sezioni e quale preprocessore  utiliz-
       zare  tramite  un'opzione della riga di comando o variabile d'ambiente.
       Se nome contiene un carattere /, allora questo viene prima provato come
       se  fosse il nome di un file, in modo da poter fare man ./foo.5 o anche
       man /cd/foo/bar.1.gz.

OPZIONI
       -C  file_configurazione
              Specifica il file man.config da usare; il valore  di  default  è
              /usr/lib/man.config (vedere man.config(5)).
```

```
lines 1-27
```

Passate alla pagina successiva con la barra spaziatrice. Potete ritornare alla pagina precedente usando il tasto **b**. Di solito, quando arrivate alla fine, **man** si interrompe e vi ritrovate al prompt. Premete **q** se volete lasciare la pagina man prima del termine o se il programma di visualizzazione non si ferma automaticamente alla fine.

Paginatori

Le combinazioni disponibili di tasti per la manipolazione delle pagine man dipendono dal *paginatore* (pager) utilizzato dalla vostra distribuzione. Molte distribuzioni usano **less** per visualizzare e scorrere avanti e indietro le pagine man. (v. **Sezione 3.3.4.2** per maggiori informazioni sui paginatori).

Ogni pagina man contiene abitualmente una coppia di sezioni standard, come possiamo notare nell'esempio di **man man**:

- La prima riga contiene il nome del comando di cui state leggendo e l'identificativo (id) della sezione in cui si trova questa pagina man. Le pagine man sono ordinate per capitoli. I comandi hanno facilmente più pagine, per esempio la pagina man della sezione utente, quella della sezione di amministratore di sistema e quella della sezione del programmatore.
- Vengono forniti il nome del comando e una breve descrizione che servono per costruire un indice di pagine man: potete cercare qualsiasi stringa nell'indice tramite il comando **apropos**.
- La sintassi del comando fornisce una annotazione tecnica di tutte le opzioni e/o argomenti accettabili. Potete pensare ad un'opzione come ad un modo di eseguire il comando. L'argomento è il destinatario dell'elaborazione. Alcuni comandi non hanno né opzioni, né argomenti. Opzioni e argomenti non necessari sono posti tra "[" e "]" per indicare che possono essere tralasciati.
- Viene fatta una descrizione più lunga del comando.
- Vengono elencate le opzioni con le loro descrizioni. Le opzioni normalmente possono essere combinate assieme: se ciò non è possibile vi viene segnalato da questa sezione.
- VARIABILI D'AMBIENTE descrive le variabili di shell che influenzano il comportamento di questo comando (non tutti ce l'hanno).
- Qualche volta ci sono delle sezioni specifiche del comando.
- La sezione "VEDERE ALTRO" contiene riferimenti ad altre pagine man. Tra parentesi c'è il numero della sezione di pagine man in cui si trova questo comando. Gli utenti esperti spesso accedono alla parte "VEDERE ALTRO" utilizzando il comando / seguito dalla stringa VED e da **Invio**.
- Normalmente ci sono anche informazioni sui bachi (bug) noti (anomalie) e su come segnalarne di nuovi da voi eventualmente riscontrati.
- Potrebbero esserci anche le informazioni circa l'autore e i diritti.

Alcuni comandi hanno numerose pagine man. Per esempio, il comando **passwd** ha una pagina man

nella sezione 1 ed un'altra nella 5. Normalmente viene mostrata la pagina man con il numero minore. Se desiderate vedere un'altra sezione rispetto a quella solita, dovete specificarla dopo il comando **man**:

man 5 *passwd*

Se invece volete vedere tutte le pagine man di un comando, una dopo l'altra, usate -a con man:

man -a *passwd*

Tale modalità, raggiunto il termine della prima pagina man e premendo di nuovo **SPAZIO**, verrà mostrata la pagina man della sezione successiva.

2.3.3. Maggiori informazioni

2.3.3.1. Le pagine Info

Oltre alle pagine man, potete leggere le pagine Info di un comando usando il comando **info**. Queste contengono di solito informazioni più recenti e sono per qualche verso più semplici da usare. Le pagine man di alcuni comandi rimandano a quelle Info.

Cominciate digitando **info *info*** in una finestra di terminale:

```
File: info.info,  Node: Top,  Next: Getting Started,  Up: (dir)

Info: An Introduction
*********************

   The GNU Project distributes most of its on-line manuals in the "Info
format", which you read using an "Info reader".  You are probably using
an Info reader to read this now.

   If you are new to the Info reader and want to learn how to use it,
type the command `h' now.  It brings you to a programmed instruction
sequence.

   To read about expert-level Info commands, type `n' twice.  This
brings you to `Info for Experts', skipping over the `Getting Started'
chapter.

* Menu:

* Getting Started::             Getting started using an Info reader.
* Expert Info::                 Info commands for experts.
* Creating an Info File::       How to make your own Info file.
* Index::                       An index of topics, commands, and variables.
--zz-Info: (info.info.bz2)Top, 24 lines --Top--------------------------------
Welcome to Info version 4.3. Type C-h for help, m for menu item.
```

Usate i tasti freccia per muovervi nel testo e per spostare il cursore su una linea che inizia con un asterisco, contenente l'argomento di cui volete informazioni, e poi premete **Invio**. Utilizzate i tasti **P** e **N** per andare all'argomento precedente o successivo . La barra spaziatrice vi porterà alla pagina successiva, senza verificare se quest'ultima inizia un nuovo argomento o la pagina info di un nuovo

comando. Impiegate **Q** per uscire. Il programma **info** ha maggiori informazioni.

2.3.3.2. I comandi whatis e apropos

Un breve indice di spiegazioni sui comandi è disponibile utilizzando il comando **whatis**, come nell'esempio qui sotto:

```
[vostro_prompt] whatis ls
ls                   (1)  - list directory contents
```

Ciò mostra una breve informazione circa un comando e la prima sezione della collezione di pagine man che contiene una pagina appropriata.

Se non sapete dove cercare e che pagina man leggere, **apropos** vi fornisce maggiori informazioni. Supponiamo che voi non sappiate come avviare un browser: potete allora battere il seguente comando:

```
un_altro_prompt> apropos browser
QDataBrowser [qdatabrowser] (3qt)  - Data manipulation and navigation for data
entry forms
QTextBrowser [qtextbrowser] (3qt)  - Rich text browser with hypertext navigation
gnome-moz-remote     (1)  - remote control of browsers
goad-browser         (1)  - Graphical GOAD browser
links                (1)  - lynx-like alternative character mode WWW browser
lynx                 (1)  - a general purpose distributed information browser
for the World Wide Web
mozilla              (1)  - a Web browser for X11 derived from Netscape
Communicator
ncftp                (1)  - Browser program for the File Transfer Protocol
```

Dopo aver premuto **Invio**, vedrete quanti argomenti relativi ai browser ci sono nella vostra macchina: non solo browser web, ma anche browser di file, FTP e di documentazione. Se avete installato i pacchetti di sviluppo, potreste anche avere le pagine man di accompagnamento concernenti programmi di scrittura che hanno a che fare con i browser. Generalmente un comando con una pagina man in sezione uno (uno rappresentato con “(1)”) può essere sperimentato in qualità di utente. Pertanto l'utente che qui sopra ha digitato **apropos** può di conseguenza provare ad avviare i comandi **links**, **lynx**, **mozilla** o **ncftp** dal momento che questi hanno a che fare con la navigazione nel c.d. world wide web.

2.3.3.3. L'opzione --help

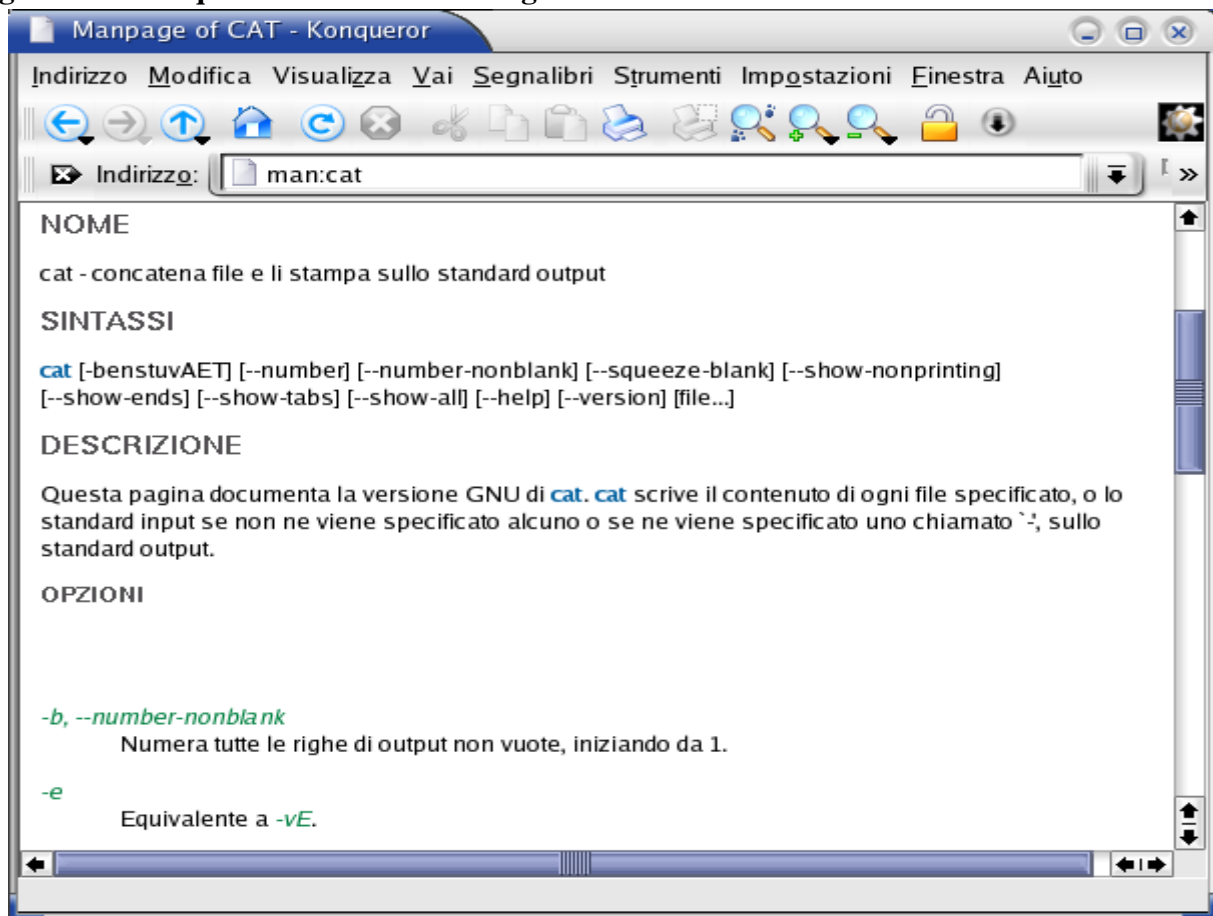
Molti comandi GNU supportano l'opzione `--help`, che dà una breve spiegazione su come usare il comando e una lista delle opzioni disponibili. Qui sotto il risultato di questa opzione con il comando **cat**:

```
prompt_utente@host: cat --help
Usage: cat [OPTION] [FILE]...
Concatenate FILE(s), or standard input, to standard output.

  -A, --show-all           equivalent to -vET
  -b, --number-nonblank    number nonblank output lines
  -e                       equivalent to -vE
  -E, --show-ends          display $ at end of each line
```

```
  -n, --number             number all output lines
  -s, --squeeze-blank      never more than one single blank line
  -t                       equivalent to -vT
  -T, --show-tabs          display TAB characters as ^I
  -u                       (ignored)
  -v, --show-nonprinting   use ^ and M- notation, except for LFD and TAB
      --help     display this help and exit
      --version  stampa le informazioni sulla versione ed esce

With no FILE, or when FILE is -, read standard input.

Report bugs to <bug-coreutils@gnu.org>.
```

2.3.3.4. Aiuti in modalità grafica

Non disperatevi se preferite un'interfaccia utente grafica (GUI). Konqueror, il normale file manager di KDE, fornisce un accesso indolore e colorato alle pagine man e Info. Potete provare “info:info” nella *barra degli indirizzi* e otterrete una pagina Info navigabile relativa al comando **info**. Allo stesso modo “man:ls” vi mostrerà la pagina man per il comando **ls**. Avete pure il completamento dei nomi dei comandi: vedrete le pagine man di tutti i comandi che iniziano con “ls” in un menu a scorrimento. Inserendo “info:/dir” nella barra degli indirizzi, appaiono tutte le pagine Info organizzate in categorie di programmi. Eccellente il contenuto di Aiuto, comprendente anche il manuale di Konqueror. Avviate quest'ultimo dal menu o digitando il comando **konqueror** in una finestra di terminale seguito da **Invio**; osservate la schermata qui sotto.

Figura 2-2. Konqueror come browser degli aiuti

Anche il browser degli aiuti Gnome è molto facile da usare. Lo potete avviare selezionando Applicazioni->Aiuto nel menu di Gnome, con un clic sull'icona del salvagente sul vostro desktop oppure inserendo il comando **gnome-help** in una finestra di terminale. La documentazione di sistema e le pagine man sono facilmente navigabili grazie alla chiara interfaccia.

Il gestore di file **nautilus** fornisce un indice di ricerca delle pagine man e info, che risultano facilmente navigabili e tra loro interconnesse. Nautilus si avvia [ndt. se installato] da linea di comando, cliccando sull'icona della vostra directory home oppure dal menu di Gnome.

Il grosso vantaggio delle GUI per la documentazione di sistema è che le informazioni sono completamente interconnesse, cosicché potete cliccare sulla sezione “VEDERE ALTRO” e dovunque appaiano collegamenti ad altre pagine man, in modo da navigare ed acquisire contemporaneamente conoscenze senza interruzioni per ore alla volta.

2.3.3.5. Eccezioni

Alcuni comandi non hanno documentazione separata poiché fanno parte di un altro comando. **cd**, **exit**, **logout** e **pwd** sono tali eccezioni: appartengono al vostro programma di shell e sono detti comandi interni alla shell. Per informazioni su questi bisogna ricorrere alle pagine man e info della vostra shell. Molti nuovi utenti hanno la shell Bash (v. Sezione 3.2.3.3 per maggiori dettagli sulle shell).

Se avete modificato la configurazione originale, potrebbe anche succedere che le pagine man, pur esistendo ancora, non siano visibili perché i vostro ambiente di shell è cambiato. In questo caso dovrete controllare la variabile `MANPATH`: la Sezione 7.2.1.2. spiega come fare ciò.

Alcuni programmi o pacchetti hanno solo un insieme di istruzioni o riferimenti nella directory `/usr/share/doc` (v. Sezione 3.3.4. per elencarli).

Nel caso peggiore, avete rimosso accidentalmente la documentazione dal vostro sistema (speriamo che sia stato proprio accidentalmente, perché è veramente una pessima idea farlo volontariamente). In tal caso, per prima cosa cercate di accertarvi che non sia rimasto ancora qualcosa utilizzando uno strumento di ricerca (v. Sezione 3.3.3.). Se è proprio così, dovete reinstallare il pacchetto che contiene il comando a cui si riferisce la documentazione (v. Sezione 7.5.).

2.4. Sommario

Abitualmente Linux funziona in modalità testo o grafica. Dal momento che oggi CPU potenti e RAM non sono più costose, ogni utente Linux può permettersi di lavorare in modalità grafica e normalmente lo fa. Ciò non significa però che voi non dobbiate conoscere il modo testo: noi lavoreremo interamente nell'ambiente testuale durante questo corso utilizzando una finestra di terminale.

Linux incoraggia i suoi utilizzatori ad acquisire conoscenze e a rendersi indipendenti. Inevitabilmente dovrete leggere molta documentazione per raggiungere questo obbiettivo: è per tale

motivo, come avrete notato, che ci riportiamo a documentazione extra per quasi ogni comando, strumento e problema elencato in questo libro. Più documenti leggerete e più semplice diverrà e più rapidamente sfoglierete manuali. Fate in modo che leggere documentazione divenga al più presto un'abitudine. Quando non sapete dare una risposta ad un problema, rivolgervi alla documentazione dovrebbe divenire una seconda natura.

Abbiamo già imparato alcuni comandi:

Tabella 2-3. Nuovi comandi nel capitolo 2: Basi

Comando	**Significato**
apropos	Cerca informazioni su un comando o argomento
cat	Mostra il contenuto di uno o più file
cd	Cambia directory
exit	Abbandona una sessione di shell
file	Informa circa il contenuto di un file
info	Legge le pagine Info di un comando
logout	Abbandona una sessione di shell
ls	Elenca il contenuto di una directory
man	Legge le pagine di manuale di un comando
passwd	Cambia la vostra password
pwd	Mostra l'attuale directory di lavoro

2.5. Esercizi

Molto di ciò che impariamo è dovuto agli errori che facciamo ed all'osservazione di come le cose possono non funzionare. Questi esercizi sono stati creati per procurarvi alcuni messaggi di errore da leggere. L'ordine in cui affronterete tali esercizi è importante.

Non scordatevi di ricorrere alle caratteristiche di Bash sulla linea di comando: tentate di svolgere gli esercizi battendo meno caratteri possibili!

2.5.1. Connessione e disconnessione

- Verificate se state lavorando in modalità testo o grafica.

  Sto lavorando in modalità testo/grafica (barrate quella sbagliata)

- Autenticatevi (login) con il nome utente e la password che avete scelto durante l'installazione.
- Disconnettetevi (logout).
- Autenticatevi nuovamente usando un nome utente inesistente.

  -> Cosa succede?

2.5.2. Password

Autenticatevi nuovamente con i vostri nome utente e password.

- Cambiate la password in *P6p3.aa!* e battete il tasto **Invio**.

  -> Cosa succede?

- Provate ancora, stavolta inserite una password ridicolmente facile, tipo *123* o *aaa*.

  -> Cosa succede?

- Provate ancora, questa volta non inserite una password ma battete soltanto il tasto **Invio**.

  -> Cosa succede?

- Provate il comando **psswd** al posto di **passwd**.

  -> Cosa succede?

Nuova password
A meno che voi non ripristiniate la vostra password a quella precedente all'esercizio, questa sarà ”P6p3.aa!”.
Mutate password dopo questo esercizio!

Notate che alcuni sistemi potrebbero non consentire di riciclare password, cioè ripristinare quella originale un certo numero di volte o un certo numero di cambi di password o ambedue.

2.5.3. Le directory

Questi sono alcuni esercizi che vi aiuteranno a capirne il senso.

- Date il comando **cd blah**

-> Cosa accade?

- Date il comando **cd ..**

  Ricordatevi lo spazio tra “cd” e “..”! Usate il comando **pwd**

  -> Cosa accade?

- Elencate il contenuto della directory con il comando **ls**.

  -> Cosa vedete?

  -> Cosa pensate che siano quelle voci?

  -> Controllate con il comando **pwd**

- Date il comando **cd**

  -> Cosa accade?

- Ripetete due volte il passo 2.

  -> Cosa accade?

- Mostrate il contenuto di questa directory.
- Provate il comando **cd `root`**

  -> Cosa accade?

  -> A quale directory avete accesso?

- Ripetete il passo 4.

  -> Conoscete un'altra maniera per sapere dove ora vi trovate?

2.5.4. I file

- Cambiate la directory a `/` e quindi a `etc`. Scrivete **ls**; se l'output è più lungo del vostro schermo, ampliate la finestra o provate **Maiuscolo+PaginaSu** e **Maiuscolo+PaginaGiu**.

  Il file `inittab` contiene la risposta alla prima domanda in questa lista. Provate il comando **file** su di questo.

  -> Il tipo di file del mio `inittab` è ......

- Usate il comando **cat `inittab`** e leggete il file.

  -> Quale è la modalità normale del vostro computer?

- Ritornate alla vostra directory home impiegando il comando **cd**
- Date il comando **file .**

  -> L'esercizio aiuta a trovare il significato di "."?
- Potete guardare "." usando il comando **cat**?
- Richiamate l'aiuto del programma **cat**, utilizzando l'opzione `--help`. Utilizzate l'opzione per numerare le linee di output per contare quanti utenti sono elencati nel file `/etc/passwd`.

2.5.5. Cercare aiuto

- Leggete **man *`intro`***
- Leggete **man *`ls`***
- Leggete **info *`passwd`***
- Date il comando **apropos `pwd`**
- Provate **man** o **info** di **cd**

  -> Come potete trovare maggiori informazioni su **cd**?

- Leggete **ls `--help`** e sperimentate.

Capitolo 3. File e file system

Dopo l'iniziale esplorazione contenuta nel Capitolo 2, siamo pronti per trattare più in dettaglio i file e le directory di un sistema Linux. Molti utenti trovano difficoltà con Linux perché mancano di una panoramica sul tipo di dati conservati in una determinata posizione. Proveremo perciò a fare un po' di luce sull'organizzazione dei file nel file system.

Elencheremo pure i file e le directory più importanti, useremo metodi diversi per visualizzare il contenuto di quei file, impareremo come possiamo creare, spostare e cancellare file e directory.

Dopo il completamente degli esercizi di questo capitolo, sarete in grado di:

- Descrivere la struttura di un file system Linux
- Mostrare e impostare percorsi (path)
- Descrivere i file più importanti, compresi kernel e shell
- Trovare file persi e nascosti
- Creare, muovere e cancellare file e directory
- Mostrare il contenuto dei file
- Comprendere ed usare tipi di collegamenti (link) differenti
- Esplorare le proprietà dei file e cambiare i loro permessi

3.1. Panoramica generale sul file system Linux

3.1.1. I file

3.1.1.1. In generale

Questa è una descrizione semplice del sistema UNIX, applicabile anche a Linux:

“In un sistema UNIX ogni cosa è un file: se qualcosa non è un file, è un processo”.

Tale affermazione è vera perché esistono file speciali che sono più di normali file (chiamati *pipe* e *socket*, per esempio) ma, per semplificare, è una generalizzazione accettabile dire che tutto è un file. Un sistema Linux, così come UNIX, non fa distinzioni tra un file e una directory dal momento che una directory è solo un file che contiene nomi di altri file. Programmi, servizi, testi, immagini, e così via, sono tutti file. Le periferiche di ingresso/uscita (ed in genere tutte le periferiche) sono considerate come file che si raccordano al sistema.

Per gestire tutti quei file in modo ordinato gli esseri umani preferiscono pensarli in una struttura organizzata a forma di albero nel disco rigido, come sappiamo ad esempio da MS-DOS (Disk Operating System). I rami principali contengono altri rami e le estremità hanno le foglie, cioè i file.

Per ora utilizzeremo questa immagine dell'albero, ma più avanti scopriremo che non si tratta di un'immagine perfettamente calzante.

3.1.1.2. Ordine dei file

Molti file sono solo file, detti *file regolari*: essi contengono dati normali, per esempio file di testo, file eseguibili o programmi, dati d'ingresso o uscita di un programma e così via.

Sebbene sia ragionevolmente sicuro ritenere che tutto ciò che incontrate in un sistema Linux sia un file, esistono tuttavia alcune eccezioni.

- *Directory*: file che sono elenchi di altri file.
- *File speciali*: il meccanismo usato per ingresso e uscita dei dati. Molti file speciali si trovano in `/dev`: ne parleremo più avanti.
- *Collegamenti* (*link*): un mezzo impiegato per rendere visibili file e directory in più parti dell'albero dei file del sistema. Tratteremo i collegamenti in dettaglio.
- *(Domain) socket*: uno speciale tipo di file, simile ai socket TCP/IP, che fornisce un'infrastruttura di processi interconnessi protetta da un controllo d'accesso del file system.
- *Named pipes*: funzionano più o meno come i socket e costituiscono un modo di comunicazione tra processi senza l'impiego della semantica dei socket di rete.

L'opzione **`-l`** di **ls** mostra il tipo di file usando il primo carattere di ciascuna linea di input:

```
jaime:~/Documents> ls -l
total 80
-rw-rw-r--  1  jaime  jaime  31744 Feb 21 17:56 intro Linux.doc
-rw-rw-r--  1  jaime  jaime  41472 Feb 21 17:56 Linux.doc
drwxrwxr-x  2  jaim   jaime   4096 Feb 25 11:50 course
```

Questa tabella offre una panoramica dei caratteri che determinano il tipo di file:

Tabella 3-1. Tipi di file in elenco esteso

Simbolo	Significato
-	File normale
d	Directory
l	Collegamento (link)
c	File speciale
s	Socket
p	Named pipe
b	Periferica a blocchi

Per non dover fare ogni volta un lungo elenco per riconoscere il tipo di file, molti sistemi normalmente non eseguono un semplice **ls**, bensì **ls `-F`**, che applica un suffisso “/=*|@” ai nomi dei file per indicarne il tipo. Per rendere la faccenda ancora più semplice per i principianti, le opzioni

`-F` e `--color` solitamente vengono combinate assieme (v. Sezione 3.3.1.1.). Noi utilizzeremo **ls -F** nel corso di questo documento per una migliore leggibilità.

Come utenti, voi avrete direttamente a che fare solo con comuni file, file eseguibili, directory e collegamenti. I file di tipo speciale esistono per consentire al sistema di fare ciò che gli richiedete e sono usati dagli amministratori e dai programmatori.

Adesso, prima di trattare dei file importanti e delle directory, dobbiamo conoscere meglio le partizioni.

3.1.2. Il partizionamento

3.1.2.1. Perché le partizioni?

Molte persone hanno una vaga idea di cosa siano le partizioni dal momento che ogni sistema operativo è capace di crearle o rimuoverle. Può sembrare strano che Linux utilizzi più di una partizione sullo stesso disco, anche quando si usa la procedura di installazione standard, così si rende necessario fornire alcune spiegazioni.

Uno degli scopi di avere diverse partizioni è quello di raggiungere un livello di sicurezza dei dati maggiore in caso di disastro. Dividendo il disco rigido in partizioni, i dati possono essere raggruppati e separati. Quando capita un incidente, solo i dati nella partizione colpita saranno danneggiati, mentre i dati nelle altre partizioni molto probabilmente sopravviveranno.

Questo principio data dai giorni in cui Linux non aveva un file system di tipo journaled e le cadute di rete elettrica potevano condurre a disastri. L'uso delle partizioni resta per motivi di sicurezza e robustezza, in modo che un guasto in una parte del sistema non significa automaticamente che l'intero computer sia in pericolo. Attualmente questa è la ragione più rilevante del partizionamento. Un semplice esempio: un utente crea uno script, un programma o un'applicazione web che inizia a riempire il disco. Se il disco contenesse solo una grande partizione, l'intero sistema potrebbe smettere di funzionare in caso di totale riempimento. Se l'utente invece conserva i dati in una partizione separata, allora solo tale partizione (dei dati) potrebbe guastarsi, mentre le partizioni di sistema e le eventuali altre di dati continuerebbero a funzionare.

Considerate che l'avere un file system “journaled” garantisce unicamente la sicurezza dei dati in caso di interruzione di rete elettrica ed improvviso scollegamento di periferiche di archiviazione. Ciò non protegge i vostri dati da blocchi guasti (bad block) ed errori logici del file system. In tali casi dovreste usare una soluzione RAID (Redundant Array of Inexpensive Disk).

3.1.2.2. Schema di partizione e tipi

Esistono due tipi di partizioni principali in un sistema Linux:

- *partizione dati*: normali dati del sistema Linux, compresa la *partizione di root* contenente tutti i dati per avviare e far funzionare il sistema;
- *partizione swap*: espansione della memoria fisica del computer, memoria extra su

disco rigido.

Molti sistemi contengono una partizione di root, una o più partizioni di dati e una o più partizioni di swap. Sistemi in ambienti misti possono avere anche partizioni per altri dati di sistema, come una partizione con file system FAT o VFAT per i dati di MS Windows.

Parecchi sistemi Linux usano **fdisk** al momento dell'installazione per impostare il tipo di partizione. Come avrete potuto notare durante l'esercizio del Capitolo 1, di solito ciò avviene automaticamente. Tuttavia in qualche occasione potreste non essere così fortunati. In tal caso dovrete sia scegliere manualmente il tipo di partizione, sia anche provvedere al partizionamento. Le partizioni standard Linux hanno i numeri 82 per swap e 83 per i dati, i quali possono essere journaled (ext3) o normali (ext2, nei sistemi più vecchi). L'utility **fdisk** ha un aiuto integrato sicché potete dimenticare questi valori.

A parte questi due, Linux supporta una varietà di altri tipi di file system, come il relativamente recente file system Reiser, JFS, NFS, FATxx e molti altri file system originariamente disponibili in altri sistemi operativi (proprietari).

La partizione di root standard (rappresentata con una singola barra "/") è di circa 100-500MB e contiene i file di configurazione del sistema, molti comandi base e programmi di server, librerie di sistema, un po' di spazio temporaneo e la directory home dell'amministratore. Una installazione standard richiede circa 250 MB per la partizione di root.

Lo spazio di swap (indicato con *swap*) è accessibile solo dal sistema stesso ed è nascosto alla vista durante le normali operazioni. Swap è il mezzo che assicura (come nei normali sistemi UNIX) che voi possiate continuare a lavorare qualsiasi cosa accada. Con Linux potenzialmente non vedrete mai irritanti messaggi tipo *Out of memory, please close some applications first and try again* (Memoria esaurita, per favore chiudere prima alcune applicazioni e riprovare), per necessità di questa memoria in più. La procedura di swap o memoria virtuale da lungo tempo adottata, solo ora lo è da parte di sistemi operativi estranei al mondo UNIX.

Usare la memoria su un disco rigido è naturalmente un'operazione più lenta rispetto all'utilizzo dei veri circuiti di memoria presenti nel computer, ma avere questa semplice funzionalità extra è una grossa comodità. Impareremo qualcosa in più sulla swap quando parleremo dei processi nel Capitolo 4.

Linux in genere si aspetta di avere il doppio della memoria fisica sotto forma di spazio di Swap sul disco rigido. Installando un sistema dovete sapere come fare ciò. Ecco un esempio su di un sistema con 512 MB di RAM:

- prima possibilità: una partizione di swap da 1 GB
- seconda possibilità: due partizioni di swap da 512 MB
- terza possibilità: con due dischi rigidi: 1 partizione da 512 MB

L'ultima opzione darà i migliori risultati quando ci si attende molte operazioni di I/O.

Leggete la documentazione del software per informazioni specifiche. Alcune applicazioni, come i database, potrebbero richiedere maggiore spazio di swap. Altri, come alcuni sistemi palmari, potrebbero non avere alcuna swap per l'assenza di un disco rigido. Lo spazio swap può dipendere anche dalla vostra versione di kernel.

Il resto del disco rigido (dischi rigidi) è generalmente diviso in partizioni di dati, sebbene potrebbe succedere che tutti i dati di sistema non critici risiedano in un'unica partizione, ad esempio quando eseguite una installazione standard per stazione di lavoro. Quando i dati non critici sono suddivisi in differenti partizioni, solitamente si segue uno schema preordinato:

- una partizione di programmi per gli utenti (*/usr*)
- una partizione contenente i dati personali degli utenti (*/home*)
- una partizione per conservare i dati temporanei come le code di stampa e di posta (*/var*)
- una partizione per software di terze parti ed extra (*/opt*)

Una volta create le partizioni, potete solo aggiungerne altre. Cambiare dimensioni o proprietà di partizioni esistenti è possibile ma non consigliabile.

La suddivisione dei dischi rigidi in partizioni è stabilita dall'amministratore di sistema: in sistemi più grandi potrebbe anche distribuire una partizione su diversi dischi rigidi utilizzando il software appropriato. Molte distribuzioni consentono impostazioni standard ottimizzate per stazioni di lavoro (utenti medi) e per impieghi come server generici, ma accettano anche partizioni personalizzate. Durante il processo di installazione potete definire il vostro schema di partizionamento sia utilizzando lo strumento specifico della vostra distribuzione, che è abitualmente una interfaccia puramente grafica, o **fdisk**, uno strumento testuale per creare partizioni e definirne le proprietà.

Una installazione workstation o cliente è destinata principalmente all'uso di una sola medesima persona. Il software selezionato per l'installazione riflette ciò ed è posta l'attenzione sui pacchetti per l'utente comune, come dei bei temi per il desktop, strumenti di sviluppo, programmi clienti per la posta elettronica, software multimediale, web ed altri servizi. Il tutto è collocato insieme su una grande partizione, viene aggiunto lo spazio di swap pari al doppio della RAM e la vostra workstation generica è completa, dotata della più vasta quantità possibile di spazio su disco per l'uso personale, ma con l'inconveniente di una possibile perdita di integrità dei dati durante situazioni critiche.

In un server i dati di sistema tendono ad essere separati da quelli degli utenti. I programmi che offrono servizi vengono tenuti in un posto diverso rispetto ai dati che essi trattano. Su questi sistemi verranno create diverse partizioni:

- una partizione con tutti i dati necessari per l'avvio della macchina
- una partizione con i dati di configurazione e i programmi server
- una o più partizioni contenenti i dati dei server come le tabelle di database, la posta degli utenti, un archivio ftp, ecc...
- una partizione con i programmi degli utenti e le applicazioni
- una o più partizioni per i file personali degli utenti (directory home)

- una o più partizioni di swap (memoria virtuale)

I server abitualmente hanno più memoria e, conseguentemente, più spazio di swap. Certi processi di server, come quelli di database, possono richiedere più spazio swap del normale: leggete la documentazione specifica per informazioni dettagliate. Per migliori prestazioni, spesso la swap è divisa in due partizioni di swap.

3.1.2.3. Punti di montaggio

Tutte le partizioni vengono agganciate al sistema tramite il punto di montaggio (*mount point*). Questo definisce la posizione di un specifico insieme di dati nel file system. Normalmente tutte le partizioni sono connesse tra loro grazie alla partizione *root* o radice. In questa partizione, che viene indicata con la barra (/), vengono create le directory. Tali directory vuote saranno il punto d'inizio delle partizioni a loro attaccate. Un esempio: data una partizione contente le seguenti directory:

```
videos/              cd−images/           pictures/
```

Vogliamo attaccare questa partizione nel file system in una directory chiamata `/opt/media`. Per fare ciò, l'amministratore di sistema deve assicurarsi che la directory `/opt/media` esista nel sistema. Preferibilmente dovrebbe essere una directory vuota. Come ciò avvenga viene spiegato più tardi in questo capitolo. Dopo, usando il comando **mount**, l'amministratore può attaccare la partizione al sistema. Quando osserverete il contenuto della directory `/opt/media` formalmente vuota, essa conterrà i file e le directory che si trovano sul dispositivo montato (disco rigido o partizione di disco rigido, CD, DVD, flash card, USB o altra unità di memorizzazione).

Durante l'avvio del sistema, tutte le partizioni vengono montate così come indicato nel file `/etc/fstab`. Alcune partizioni non vengono montate in automatico, per esempio se non sono collegate costantemente al sistema, come le memorie usate dalla vostra macchina fotografica digitale. Se ben configurata, la periferica sarà montata non appena il sistema si accorgerà che è collegata, oppure può essere montabile dall'utente, cioè non avete bisogno di essere amministratore di sistema per attaccare e staccare la periferica al/dal sistema. C'è un esempio nella Sezione 9.3.

In un sistema funzionante le informazioni sulle partizioni e sui relativi punti di montaggi possono essere mostrate usando il comando **df** (che significa *disk full* o *disk free*). In Linux, **df** è la versione GNU e supporta l'opzione `-h` o *human readable* (leggibile da esseri umani) che migliora notevolmente la leggibilità. Notate che le macchine UNIX commerciali hanno comunemente le loro versioni particolari di **df** e di molti altri comandi. Il loro comportamento di solito è lo stesso, sebbene le versioni GNU degli strumenti comuni spesso hanno funzioni migliori e più numerose.

Il comando **df** mostra solo informazioni circa le partizioni attive non di swap. Queste possono comprendere partizioni da altri sistemi in rete, come nell'esempio qui sotto dove le directory home sono montate da un file server sulla rete, situazione che si incontra spesso in ambienti aziendali.

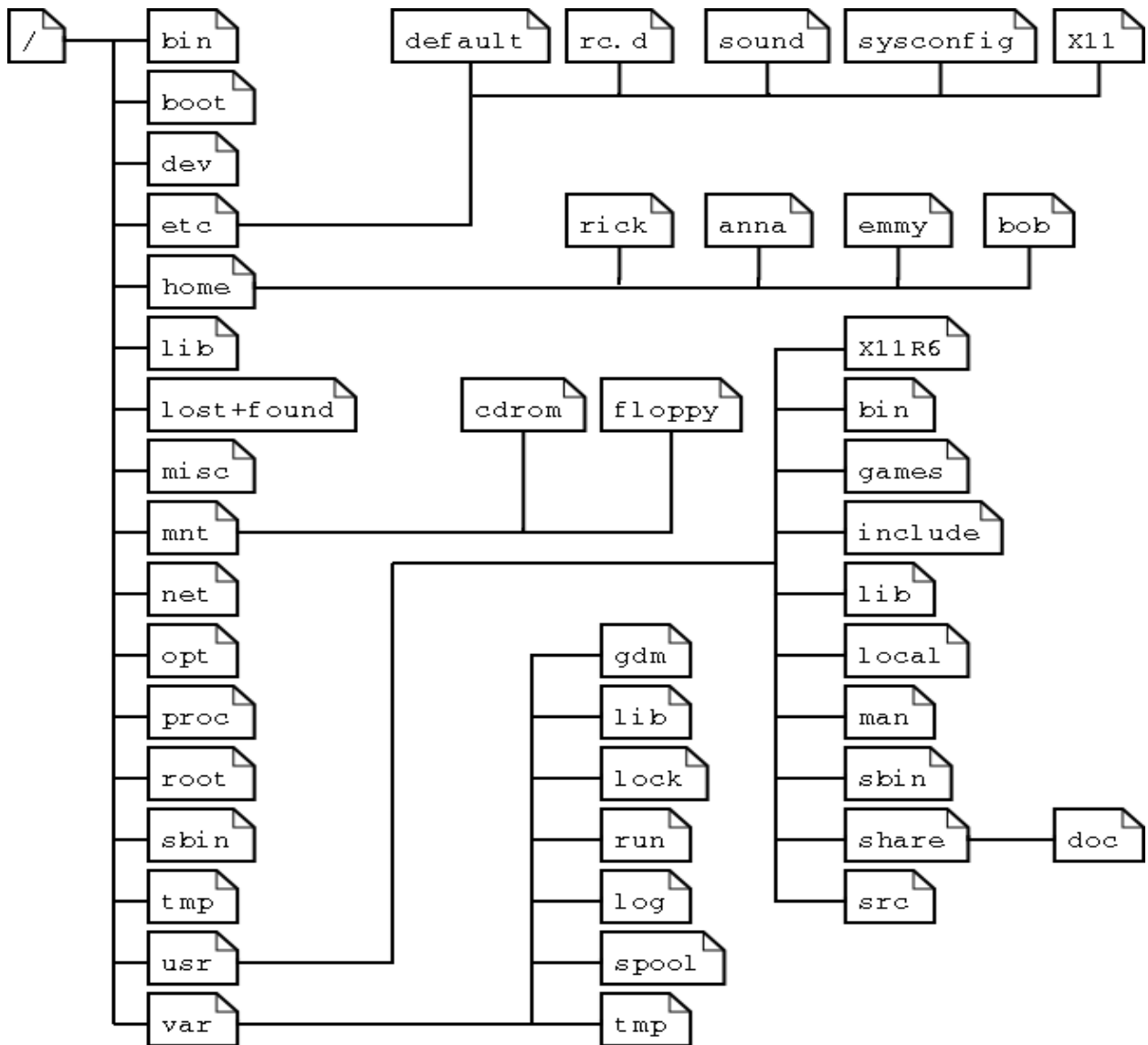
```
freddy:~> df -h
Filesystem          Size  Used  Avail  Use%  Mounted on
/dev/hda8           496M  183M   288M   39%  /
/dev/hda1           124M  8.4M   109M    8%  /boot
/dev/hda5            19G   15G   2.7G   85%  /opt
/dev/hda6           7.0G  5.4G   1.2G   81%  /usr
```

```
/dev/hda7              3.7G  2.7G   867M   77%  /var
fs1:/home              8.9G  3.7G   4.7G   44%  /.automount/fs1/root/home
```

3.1.3. Di più sulla struttura del file system

3.1.3.1. Visuale

Per comodità il file system Linux è abitualmente immaginato come una struttura ad albero. In un sistema Linux standard troverete una struttura simile allo schema presentato di seguito.

Figura 3-1. Struttura del file system Linux

Questa è la struttura di un sistema RedHat. A seconda dell'amministratore di sistema, il sistema operativo e la missione della macchina UNIX, la struttura può cambiare e le directory possono

mancare oppure essere aggiunte a piacere. I nomi non sono neanche richiesti: sono solo una convenzione.

L'albero del file system inizia dal tronco o sbarra, rappresentato da una sbarra (/). Tale directory, contenente tutti i file e le directory sottostanti, è chiamata *root directory* (*directory radice*) o "*radice*" del file system.

Le directory che si trovano solo ad un livello al di sotto di quella di root sono spesso precedute da una sbarra (*slash*) per indicare la loro posizione e per evitare di confonderle con altre directory che potrebbero avere lo stesso nome. Quando si inizia con un nuovo sistema è sempre una buona idea dare un'occhiata alla directory radice. Vediamo come potete accedervi:

```
emmy:~> cd /
emmy:~> ls
bin/   dev/  home/    lib/         misc/  opt/   root/  tmp/ var/
boot/  etc/  initrd/  lost+found/  mnt/   proc/  sbin/  usr/
```

Tabella 3-2. Sottodirectory della directory radice

Directory	Contenuto
/bin	Comuni programmi condivisi dal sistema, dall'amministratore di sistema e dagli utenti.
/boot	I file di avvio e del kernel, `vmlinuz`. In alcune recenti distribuzioni ci sono anche i dati di `grub`. Grub è il GRand Unified Boot loader e rappresenta un tentativo di sbarazzarsi di molti boot-loader differenti che conosciamo oggi.
/dev	Contiene i riferimenti a tutto l'hardware periferico della CPU, che viene rappresentato come file con particolari caratteristiche.
/etc	In `/etc` si trovano file di configurazione di sistema molto importanti. Questa directory contiene dati simili a quelli del Pannello di Controllo di Windows
/home	Directory home [personale] dei normali utenti.
/initrd	(in alcune distribuzioni) Informazioni per l'avvio. Non rimuovere!
/lib	File di libreria, comprende file per tutti i tipi di programmi necessari al sistema ed agli utenti.
/lost+found	Ogni partizione possiede `lost+found` nella directory più alta: vi si trovano i file salvati durante un guasto.
/misc	Per usi diversi.
/mnt	Punto di montaggio standard per file system esterni, ad es. un CD-ROM o una fotocamera digitale.
/net	Punto di montaggio standard per interi file system remoti.
/opt	Contiene tipicamente software extra e di terze parti.

Directory	Contenuto
/proc	Un file system virtuale contenente informazioni sulle risorse di sistema. Maggiori informazioni sul significato dei file in `proc` si ottengono dando il comando **man** ***proc*** in una finestra di terminale. Il file `proc.txt` tratta in dettaglio del file system virtuale.
/root	La directory home dell'utente amministratore. Tenete presente la differenza tra /, la directory root o radice, e /root, la home directory dell'utente *root*.
/sbin	Programmi utilizzati dal sistema e dall'amministratore di sistema.
/tmp	Spazio temporaneo usato dal sistema, ripulito ad ogni riavvio: da non usare per salvare alcun lavoro!
/usr	Programmi, librerie, documentazione, ecc... per tutti i programmi a disposizione degli utenti.
/var	Deposito di tutti i file variabili e temporanei creati dagli utenti, come i file di log, le code di posta, l'area per lo spooler di stampa, spazio per l'archiviazione temporanea dei file scaricati da internet, o per conservare l'immagine di un CD prima di masterizzarlo.

Come si può scoprire in quale partizione si trova una directory? Usando il comando **df** con un punto (.) come opzione mostra la partizione a cui appartiene la directory corrente e informa sulla quantità di spazio usato in tale partizione:

```
sandra:/lib> df -h .
Filesystem           Size   Used  Avail  Use%  Mounted on
/dev/hda7            980M   163M   767M   18%  /
```

Come regola generale, ogni directory sotto la directory root si trova nella partizione di root, a meno che non esista una voce separata nell'elenco completo prodotto da **df** (o **df -h** senza altre opzioni).

Leggete di più in **man** ***hier***.

3.1.3.2. Il file system in realtà

Per molti utenti e per molti compiti comuni degli amministratori di sistema è sufficiente sapere che i file e le directory sono ordinati in una struttura ad albero. Il computer, comunque, non capisce nulla di alberi e di strutture ad albero.

Ogni partizione ha un proprio file system. Immaginando tutti quei file system insieme, possiamo farci un idea della struttura ad albero dell'intero sistema, ma non è così semplice come quella. In un file system, un file è rappresentato da un *inode*, un tipo di numero seriale che contiene informazioni sui dati veri che costituiscono il file, su chi possiede il file e dove questo si trova nel disco rigido.

Ogni partizione ha il suo proprio insieme di inode: grazie ad un sistema con partizioni multiple possono esistere file con lo stesso numero di inode.

Ciascun inode descrive una struttura di dati nel disco rigido, conservando le proprietà di un file, compresa la locazione fisica dei dati del file. Quando si prepara un disco fisso per accettare l'archiviazione dei dati, normalmente durante l'iniziale processo di installazione del sistema, si crea un numero fisso di inode per partizione. Questo numero sarà la quantità massima di file di tutti i tipi (comprese directory, file speciali, collegamenti, ecc...) che potranno esistere contemporaneamente nella partizione. Normalmente contiamo di avere 1 inode da 2 a 8 kilobyte di memoria.

Ogni qualvolta si crea un nuovo file, questo ottiene un inode libero contenente le seguenti informazioni:

- Proprietario (*owner*) e gruppo possessore del file.
- Tipo del file (normale, directory, ecc...)
- Permessi sul file (v. Sezione 3.4.1.)
- Data ed ora di creazione, ultima lettura e modifica.
- Data ed ora in cui tale informazione è stata cambiata nell'inode.
- Numero di collegamenti a questo file (v. più oltre in questo capitolo).
- Dimensione del file
- Un indirizzo che definisce la vera posizione dei dati del file.

L'unica informazione non inclusa in un inode è il nome del file e della directory. Questi sono conservati in speciali file di directory. Confrontando i nomi dei file e i numeri di inode, il sistema è in grado di costruire una struttura ad albero comprensibile per l'utente. Gli utenti possono vedere i numeri di inode utilizzando l'opzione `-i` con **ls**. Gli inode dispongono di un proprio spazio separato nel disco.

3.2. Orientarsi nel file system

3.2.1. Il percorso

Quando volete che il sistema esegua un comando, non dovete quasi mai dare il percorso (*path*) completo di quel comando. Per esempio, sappiamo che il comando **ls** si trova nella directory `/bin` (controllate con **which -a** ***ls***), cosicché non dobbiamo inserire il comando **/bin/ls** affinché il computer elenchi il contenuto della corrente directory.

La variabile d'ambiente `PATH` si occupa di ciò. Tale variabile elenca le directory del file system in cui si possono trovare i file eseguibili e così risparmia all'utente molte battiture di caratteri e memorizzazioni delle posizioni dei comandi. Così `PATH` contiene solitamente molte directory aventi `bin` da qualche parte nel loro nome, come mostra l'utente qui sotto. Il comando **echo** si usa per vedere il contenuto (“$”) della variabile `PATH`:

```
rogier:> echo $PATH
/opt/local/bin:/usr/X11R6/bin:/usr/bin:/usr/sbin:/bin
```

In questo esempio, le directory `/opt/local/bin`, `/usr/X11R6/bin`, `/usr/sbin` , `/usr/sbin` e `/bin` vengono scandite alla ricerca del programma richiesto. Non appena c'è una

corrispondenza, la ricerca viene terminata, anche se non tutte le directory nel percorso sono state passate. Ciò può portare a strane situazioni. Nel primo esempio qui di seguito l'utente sa che esiste un programma chiamato **sendsms** per inviare un messaggio SMS e che un altro utente dello stesso sistema può usarlo, ma lui no. La differenza consiste nella configurazione della variabile `PATH`:

```
[jenny@blob jenny]$ sendsms
bash: sendsms: command not found
[jenny@blob jenny]$ echo $PATH
/bin:/usr/bin:/ust/bin/X11:/usr/X11R6/bin:/home/jenny/bin
[jenny@blob jenny]$ su - tony
Password:
tony:~>which sendsms
sendsms is /usr/local/bin/sendsms

tony:~>echo $PATH
/home/tony/bin.Linux:/home/tony/bin:/usr/local/bin:/usr/local/sbin:\
/usr/X11R6/bin:/usr/bin/:/usr/sbin:/bin:/sbin
```

Notate l'uso di **su** (switch user), che vi consente di eseguire una shell nell'ambiente di un altro utente a condizione che ne conosciate la password.

Una sbarra rovescia indica la continuazione di una linea su quella successiva senza l'interruzione di un **Invio** da una linea all'altra.

Nel prossimo esempio un utente vuole richiamare il comando **wc** (word count) per controllare il numero di linee in un file, ma non accade nulla e perciò deve interrompere tale azione usando la combinazione **Ctrl+C**:

```
jumper:~> wc -l test

(Ctrl-C)
jumper:~> which wc
wc is hashed (/home/jumper/bin/wc)

jumper:~> echo $PATH
/home/jumper/bin:/usr/local/bin:/usr/local/sbin:/usr/X11R6/bin:\
/usr/bin:/usr/sbin:/bin:/sbin
```

L'uso del comando **which** ci mostra che questo utente ha una directory `bin` nella sua home directory, contenente un programma che è anche chiamato **wc**. Dal momento che il programma nella sua home directory viene trovato per primo mentre si stanno scorrendo i percorsi alla ricerca di **wc**, tale programma “fatto in casa” viene eseguito con un input che probabilmente non è in grado di comprendere, cosicché siamo costretti ad interromperlo. Esistono vari modi per risolvere questo problema (ci sono sempre diversi modi per risolvere un problema in UNIX/Linux): una risposta potrebbe essere quella di rinominare il programma **wc** dell'utente, oppure l'utente potrebbe dare il percorso completo fino al comando desiderato, che può essere trovato utilizzando `-a` con il comando **which**.

Se l'utente utilizza più frequentemente programmi in altre directory, può cambiare il suo percorso per cercare per ultimo nelle sue directory:

```
jumper:~> export PATH=/usr/local/bin:/usr/local/sbin:/usr/X11R6/bin:\
/usr/bin:/usr/sbin:/bin:/sbin:/home/jumper/bin
```

I cambiamenti non sono permanenti!

Notate che, quando si usa il comando **export** in una shell, i cambiamenti sono temporanei e validi solo per la sessione in corso (finché non la chiudete). L'apertura di nuove sessioni, anche mentre quella corrente è ancora attiva, non vi creerà un nuovo percorso in esse. Vedremo nella Sezione 7.2 come possiamo rendere permanenti nell'ambiente questo tipo di modifiche, aggiungendo quelle linee ai file di configurazione della shell.

3.2.2. Percorsi assoluti e relativi

Un percorso (*path*), cioè la strada da seguire nella struttura ad albero per raggiungere un certo file, può essere descritto partendo dal tronco dell'albero (la "/" o directory radice). In tal caso, il percorso inizia con una sbarra e viene chiamato percorso assoluto, dal momento che non possono esserci errori: nel sistema solo un file potrà corrispondervi.

Nell'altro caso, il percorso non inizia con una sbarra ed è possibile confondere tra `~/bin/wc` (nella directory home dell'utente) e `bin/wc` in `/usr` come nell'esempio precedente. I percorsi che non iniziano con una sbarra sono sempre relativi.

Nei percorsi relativi usiamo anche . e .. per indicare la directory corrente e quella genitrice. Una coppia di esempi pratici:

- Quando volete compilare del codice sorgente, spesso la documentazione di installazione vi dice di avviare il comando **./configure**, che lancia il programma *configure* collocato nella directory corrente (creata con il nuovo codice) invece di eseguire un altro programma configure presente altrove nel sistema.
- Nei file HTML i percorsi relativi sono spesso utilizzati per creare un insieme di pagine facilmente trasferibili in altri posti:
```
<img alt="Garden with trees" src="../images/garden.jpg">
```
- Notate la differenza ancora una volta:
```
theo:~> ls /mp3
ls: /mp3: No such file or directory
theo:~> ls mp3/
oriental/  pop/  sixties/
```

3.2.3. I file e le directory più importanti

3.2.3.1. Il kernel

Il kernel è il cuore del sistema: gestisce le comunicazioni tra l'hardware sottostante e le periferiche. Il kernel assicura inoltre che processi e daemon (demoni o processi server) vengano avviati e fermati al momento giusto. Ha una quantità così grande di altri compiti importanti che esiste una speciale mailing list, dedicata allo suo sviluppo, che si occupa solo di questo argomento e dove vengono condivise montagne di informazioni. Dibattere in dettaglio sul kernel ci porterebbe troppo

lontano: per ora è sufficiente sapere che il kernel è il file più importante del sistema.

3.2.3.2. La shell

3.2.3.2.1. Cos'è una shell?

Cercare una spiegazione appropriata del concetto di *shell* mi ha creato più problemi di quanti mi attendessi. Sono disponibili tutti i generi di spiegazioni, che vanno dal semplice paragone come "la shell è lo sterzo di un'auto", alla vaga definizione del manuale di Bash che afferma che "bash è un interprete del linguaggio a comandi compatibile sh" o ad altre ancora più oscure espressioni come "una shell gestisce l'interazione tra il sistema e i suoi utenti". Una shell è molto più di questo.

Una shell può essere meglio paragonata ad un mezzo per parlare con il computer, un linguaggio. Molti utenti conoscono quell'altro linguaggio, il linguaggio "punta-e-clicca" del desktop, ma con esso il computer guida la conversazione, mentre l'utente assume il ruolo passivo di selezionare le funzioni che questo gli presenta. E' molto difficile per un programmatore includere tutte le opzioni e gli usi possibili di un comando in un formato GUI. Così le GUI sono quasi sempre meno potenti del comando o dei comandi che formano il backend [cioè su cui queste si appoggiano per svolgere le varie funzioni].

La shell, d'altro canto, è una maniera evoluta di dialogare con il sistema poiché consente la conversazione nei due sensi e di assumere l'iniziativa. Entrambe le parti della comunicazione sono uguali, così si possono sperimentare nuove idee. La shell permette all'utente di gestire un sistema in modo piuttosto flessibile e, come ulteriore vantaggio, di automatizzare i compiti.

3.2.3.2.2. Tipi di shell

Come la gente conosce linguaggi e dialetti differenti, così il computer conosce tipi diversi di shell:

- **sh** o Bourne Shell: la shell originale ancora utilizzata su sistemi UNIX e in ambienti collegati a UNIX. E' la comune shell, un piccolo programma con poche funzioni. Quando è in modalità POSIX-compatibile **bash** emula questa shell.
- **bash** o Bourne Again SHell: la shell GNU standard, intuitiva e flessibile. Probabilmente molto consigliabile sia ai principianti sia, contemporaneamente, agli esperti e professionisti essendo uno strumento molto potente. In Linux **bash** è la shell standard per i comuni utenti. Tale shell viene anche detta *superinsieme* della shell Bourne, un insieme di aggiunte e plug-in. Ciò significa che la shell Bourne Again è compatibile con quella Bourne: i comandi che funzionano con **sh**, funzionano pure con **bash**. Comunque non è sempre vero il contrario. Tutti gli esempi e gli esercizi di questo libro usano **bash**.
- **csh** o C Shell: la sintassi di questa shell assomiglia a quella del linguaggio di programmazione C. Qualche volta viene richiesta dai programmatori.
- **tcsh** o Turbo C Shell: un superinsieme della comune C shell, che aumenta la semplicità e la velocità.
- **ksh** o Korn shell: apprezzata qualche volta da persone con esperienze di UNIX. E' un superinsieme della Bourne shell: in configurazione base è un incubo per i

principianti.

Il file `/etc/shells` offre una panoramica delle shell conosciute da un sistema Linux:

```
mia:~> cat /etc/shells
/bin/bash
/bin/sh
/bin/tcsh
/bin/csh
```

Bourne shell falsa

Notate che normalmente `/bin/sh` è un collegamento a Bash, che, se chiamata in questo modo, verrà eseguita in modalità Bourne shell.

La vostra shell di base è impostata nel file `/etc/passwd`, come in questa linea che riguarda l'utente *mia*:

```
mia:L2NOfqdlPrHwE:504:504:Mia Maya:/home/mia:/bin/bash
```

Per passare da una shell all'altra, basta solo inserire il nome della nuova shell nel terminale attivo. Il sistema trova la directory dove ricorre il nome usando le impostazioni di `PATH` e, dal momento che una shell è un file eseguibile (programma), la shell corrente la attiva e la manda in esecuzione. Abitualmente appare un nuovo prompt perché ogni shell ha il proprio aspetto tipico:

```
mia:~> tcsh
[mia@post21 ~]$
```

3.2.3.2.3. Quale shell sto usando?

Se non sapete quale shell state utilizzando, potete sia controllare la linea del vostro account in `/etc/passwd`, sia dare **echo $SHELL** come comando.

3.2.3.3. La vostra home directory

Quando vi connettete al sistema, la vostra home directory [directory personale] è la destinazione iniziale. In molti casi si tratta di una sottodirectory di `/home`, sebbene possa variare. La vostra home directory può essere collocata nel disco rigido di un file server remoto: in questo caso essa si può trovare in `/nethome/vostronomeutente`. In caso diverso l'amministratore di sistema può aver optato per una struttura meno comprensibile e la vostra directory, ad esempio, potrebbe essere su `/disk6/HU/07/jgillard`.

Non dovete preoccuparvi eccessivamente di quale sia il percorso della vostra home directory: infatti il percorso corretto è conservato nella variabile d'ambiente `HOME`, nel caso alcuni programmi ne abbiano bisogno. Con il comando **echo** potete vedere il contenuto di questa variabile:

```
orlando:~> echo $HOME
/net/home/orlando
```

Nella vostra directory personale potete fare tutto ciò che vi piace: potete metterci tanti file in tante

directory quanti ne volete, sebbene lo spazio totale per dati e i file sia naturalmente limitato a causa dell'hardware, delle dimensioni delle partizioni e, alle volte, dall'applicazione di un sistema di quota da parte dell'amministratore. Era infatti pratica comune limitare l'uso del disco quando i dischi rigidi erano ancora costosi. Al giorno d'oggi i limiti vengono applicati quasi esclusivamente in grossi ambienti. Potete verificare da voi stessi l'esistenza di un limite utilizzando il comando **quota**:

```
pierre@lamaison:/> quota -v
Disquotas for user pierre (uid 501): none
```

Nel caso siano state stabilite delle quote, otterrete una lista delle partizioni limitate e delle loro specifiche limitazioni. L'eccedere i limiti può essere comunque tollerato per un breve periodo con alcune minime o nulle restrizioni. Informazioni dettagliate si possono trovare usando i comandi **info** ***quota*** o **man** ***quota***.

Nessuna Quota?

Se il vostro sistema non riesce a trovare **quota**, allora significa che non sono state applicate limitazioni all'uso del file system.

La vostra directory personale viene indicata con una tilde (~), abbreviazione di `/path_to_home/nome_utente`. Lo stesso percorso è conservato nella variabile `HOME`, cosicché non dovete fare nulla per attivarlo. Una semplice applicazione: passare da `/var/music/albums/arno/2001` a `images` contenuta nella vostra directory personale utilizzando un comando elegante:

```
rom:/var/music/albums/arno/2001> cd ~/images

rom:~/images> pwd
/home/rom/images
```

Più avanti in questo capitolo discuteremo dei comandi per la gestione dei file e delle directory allo scopo di mantenere ordinata la vostra directory personale.

3.2.4. I file di configurazione più importanti

Come abbiamo accennato in precedenza, molti file di configurazione sono riposti nella directory `/etc`. Il contenuto può essere visualizzato ricorrendo al comando **cat**, che invia i file testuali allo standard output (normalmente il vostro monitor). La sintassi è la seguente:

cat **`file1 file2 ... fileN`**

In questa sezione proveremo ad offrire una panoramica dei file di configurazione più comuni. Questo non è un elenco esaustivo: installando pacchetti extra si possono aggiungere altri file di configurazione in `/etc`. Leggendo i file di configurazione scoprirete che solitamente essi sono abbastanza ben commentati ed autoesplicanti. Alcuni file hanno anche delle pagine man che contengono documentazione extra, come **man** ***group***.

Tabella 3-3. I più comuni file di configurazione

File	Informazioni/servizio
aliases	Il file degli alias di posta utilizzato dai server di posta Sendmail e Postfix. L'uso di un mail server nel mondo UNIX è da lunghi anni comune in tutti i sistemi e quasi ogni distribuzione Linux contiene ancora un pacchetto Sendmail. In questo file i nomi dell'utente locale vengono accoppiati con i nomi reali, quando essi ricorrono negli indirizzi E-mail, o con altri indirizzi locali.
apache	I file di configurazione del web server Apache
bashrc	Il file di configurazione globale della Bourne Again SHell. Definisce le funzioni e gli alias per tutti gli utenti. Le altre shell possono avere il proprio file di configurazione globale, come cshrc.
crontab e le directory cron.*	Configurazione delle operazioni da eseguirsi periodicamente - backup, aggiornamenti dei database di sistema, pulizia del sistema, rotazione dei log, ecc...
default	Le opzioni di base per certi comandi, come per **useradd**.
filesystems	File system noti: ext3, vfat, iso9660, ecc...
fstab	Elenca le partizioni con i relativi punti di montaggio (*mount point*).
ftp*	Configurazione del server ftp: chi può connettersi, quali parti del sistema sono accessibili, ecc...
group	File di configurazione dei gruppi di utenti. Utilizzate le utility ombra **groupadd**, **groupmod** e **groupdel** per modificare questo file. Modificatelo manualmente solo se realmente sapete cosa state facendo.
hosts	Un elenco di macchine che possono essere contattate tramite rete, ma senza la necessità di un servizio di risoluzione dei nomi di dominio (DNS). Questo non ha nulla a che fare con la configurazione di rete del sistema, che si attua con /etc/sysconfig.
inittab	Informazioni per l'avvio: modalità, numero delle console di testo, ecc...
issue	Informazioni sulla distribuzione (versione di rilascio e/o info sul kernel)
ld.so.conf	Locazioni dei file di libreria
lilo.conf, silo.conf, aboot.conf, ecc...	Informazioni di avvio per il LInux LOader, il sistema di avvio che ora gradualmente sarà soppiantato da GRUB.
logrotate.*	Rotazione dei log, un sistema che previene la raccolta di una quantità eccessiva di file di log.
mail	Directory contenente istruzioni per il comportamento del mail server.
modules.conf	Configurazione dei moduli che abilitano funzioni speciali (driver).
motd	Message Of The Day: mostrato a chiunque si colleghi al sistema (in modo testuale), può essere usato dall'amministratore di sistema per annunciare servizi/manutenzioni ecc...
mtab	File system correntemente montati. Si avvisa di non modificare mai questo file.
nsswitch.conf	Ordine secondo cui contattare i risolutori dei nomi quando un processo richiede la risoluzione di un nome di host.
pam.d	Configurazione dei moduli di autenticazione
passwd	Elenca gli utenti locali. Utilizzate le utility ombra **useradd**, **usermod** e **userdel** per modificare questo file. Modificatelo manualmente solo quando sapete veramente cosa state facendo.

File	Informazioni/servizio
printcap	File di configurazione di stampa sorpassato ma spesso ancora frequentemente utilizzato. Non modificatelo manualmente a meno che non sappiate realmente cosa state facendo.
profile	Estesa configurazione di sistema dell'ambiente di shell: variabili, proprietà basiche dei nuovi file, limitazioni delle risorse, ecc...
rc*	Directory che definiscono i servizi attivi per ogni run level.
resolv.conf	Ordine da seguire per contattare i server DNS (solo Domain Name Server).
sendmail.cf	Principale file di configurazione del server Sendmail.
sndconfig o sound	Configurazione della scheda sonora e degli eventi sonori.
ssh	Directory contenente i file di configurazione del client e del server secure shell
sysconfig	Directory contenente i file di configurazione del sistema: mouse, tastiera, rete, desktop, orologio di sistema, gestione dell'alimentazione elettrica, ecc... (specifico di RedHat).
X11	Impostazioni del server grafico X. RedHat usa Xfree che è richiamato nel nome del principale file di configurazione, Xfree86Config. Contiene inoltre le direttive generali per i window manager disponibili nel sistema, per esempio **gdm**, **fvwm**, **twm**, ecc...
xinetd.* o inetd.conf	I file di configurazione dei servizi Internet avviati dal daemon di sistema dei servizi (estesi) Internet (server che non avviano daemon indipendenti).

Nel corso di questa guida impareremo di più su questi file e ne studieremo alcuni in dettaglio.

3.2.5. I più comuni *device*

I device (dispositivi), in genere ogni unità periferica di un PC che non sia la CPU stessa, vengono presentati al sistema come voci della directory /dev. Uno dei vantaggi di questo approccio UNIX di gestione delle periferiche è che né l'utente, né il sistema devono preoccuparsi troppo delle caratteristiche di queste.

In genere gli utenti novizi di Linux o UNIX sono spesso sopraffatti dalla quantità di nuovi nomi e concetti che sono costretti ad imparare. Ecco il perché in questa introduzione è stato accluso un elenco dei comuni dispositivi.

Tabella 3-4. Comuni periferiche

Nome	Device o dispositivo
cdrom	CD drive
console	Voce speciale per la console correntemente in uso.
cua*	Porte seriali
dsp*	Periferiche per il campionamento e la registrazione
fd*	Nomi della maggioranza di tipi di floppy drive: quello base è /dev/fd0, un lettore per floppy da 1.44 MB.

Nome	Device o dispositivo
hd[a-t][1-16]	Supporto standard per periferiche IDE con la capacità massima di partizioni per ciascuna
ir*	Periferiche all'infrarosso
isdn*	Gestione delle connessioni ISDN
js*	Joystick
lp*	Stampanti
mem	Memoria
midi*	Lettori midi
mixer* e music	Modello ideale di un mixer (combina o somma segnali)
modem	Modem
mouse (anche msmouse, logimouse, psmouse, input/mice, psaux)	Tutti i tipi di mouse
null	Bidone della spazzatura senza fondo
par*	Nomi del supporto delle porte parallele
pty*	Pseudo terminali
radio*	Per radioamatori (HAM)
ram*	Device di avvio
sd*	Dischi SCSI con le loro partizioni
sequencer	Per applicazioni audio che usano le caratteristiche di sintetizzazione della scheda sonora (controllore di periferiche MIDI)
tty*	Console virtuali che emulano i terminali vt100
usb*	Periferiche USB (schede, scanner, ecc...)
video*	Per l'uso di una scheda grafica con supporto video.

3.2.6. I più comuni file di variabili

Nella directory /var troviamo un insieme di variabili per la conservazione di dati specifici non-costanti (al contrario del programma ls o dei file di configurazione del sistema che cambiano poche volte o addirittura mai). Tutti i file che cambiano frequentemente, come quelli di log, le caselle postali, i lock file, gli spooler, ecc.., vengono tenuti in una sottodirectory di /var.

Come misura di sicurezza questi file sono tenuti normalmente in luoghi separati dai principali file di sistema, cosicché siamo in grado di osservarli meglio e di impostare, se necessario, permessi più stretti. Molti di questi file hanno pure bisogno di permessi più estesi del solito, come /var/tmp, che necessita di essere scrivibile da chiunque. Molte attività dell'utente potrebbero svolgersi colà, generate anche da anonimi utenti Internet connessi al vostro sistema. Questo è uno dei motivi per cui la directory /var, comprese tutte le sue sottodirectory, si trova in una partizione separata. In questo modo, per esempio non sussiste il rischio che una mail bomb, invada il resto del sistema operativo che contiene dati più importanti, come i vostri programmi e i file di configurazione.

/var/tmp/ e /tmp

I file in `/tmp` possono essere cancellati senza preavviso da regolari operazioni di sistema o a causa di un riavvio dello stesso. In alcuni sistemi (personalizzati) anche `/var/tmp` potrebbe comportarsi in modo imprevedibile. Tuttavia, dal momento che non è il caso normale, vi suggeriamo di usare la directory `/var/tmp` per il salvataggio dei file temporanei. In caso di dubbi, controllate con il vostro amministratore di sistema. Se gestite voi il vostro sistema, potrete essere ragionevolmente certi che questo è un luogo sicuro se non avete cambiato le impostazioni di `/var/tmp` (come root, in quanto un utente normale non può farlo).

Qualsiasi cosa facciate, provate a mantenere i privilegi garantiti ad un utente normale - non cominciate a salvare file direttamente nella directory radice (/) del file system, non metteteli in `/usr` o in altre sottodirectory o in un altro posto riservato. Ciò limita notevolmente il vostro accesso ai file system sicuri.

Uno dei principali sistemi di sicurezza di UNIX, che naturalmente è stato bene implementato su tutte le macchine Linux, è la funzione di conservazione dei log, che registra tutte le azioni dell'utente, i processi, gli eventi di sistema, ecc... Il file di configurazione del cosiddetto *syslogdaemon* stabilisce quale informazione e per quanto essa sarà mantenuta registrata. La posizione originale di tutti i log è `/var/log`, che contiene file differenti per i log d'accesso, dei server, per i messaggi di sistema, ecc...

In `/var` troviamo di solito i dati dei server, mantenuti qui per separarli dai dati critici come il programma server stesso e i suoi file di configurazione. Un tipico esempio in Linux è `/var/www`, che contiene le pagine HTML attive, script e immagini che offre un web server. L'albero FTP di un server FTP (dati che possono essere scaricati da un cliente remoto) è altresì ben conservato in una delle sottodirectory di `/var`. Siccome tali dati sono pubblicamente accessibili e spesso modificabili da utenti anonimi, è più sicuro tenerli qui, lontani da partizioni o directory con dati sensibili.

In molte installazioni di tipo stazione di lavoro, `/var/spool` avrà almeno le directory `at` e `cron`, contenenti le operazioni programmate. In ambienti d'ufficio tale directory contiene `lpd`, che tiene sia le code di stampa ed altri file di configurazione della stampante, sia i file dei log di quest'ultima.

In sistemi server troveremo in genere `/var/spool/mail`, contenente la posta in arrivo degli utenti locali, ordinata in un file per utente, la “inbox” dell'utente. Una directory correlata è `mqueue`, l'area di spooler per i messaggi di posta non inviati. Queste parti del sistema sono molto occupate nei server di posta con molti utenti. Anche i news server usano l'area `/var/spool` a causa dell'enorme quantità di messaggi da elaborare.

La directory `/var/lib/rpm` è specifica delle distribuzioni basate su RPM (RedHat Package Manager): è l'area dove sono conservate le informazioni dei pacchetti RPM. Anche altri gestori di pacchetti solitamente salvano i loro dati da qualche parte in `/var`.

3.3. Manipolare i file

3.3.1. Vedere le proprietà dei file

3.3.1.1. Di più su ls

Come abbiamo già detto, oltre al nome del file, **ls** può fornire una quantità di ulteriori informazioni, ad esempio il tipo di file. E' inoltre capace di mostrare i permessi di un file, la sua dimensione, il numero di inode, data e ora di creazione, proprietari e quantità di collegamenti ad esso. Con l'opzione `-a` di **ls** possono essere visualizzati i file che normalmente sono nascosti alla vista. Questi sono i file che hanno il nome iniziante con un punto. Una coppia di esempi tipici include i file di configurazione della vostra home directory. Se avete lavorato per un po' con un certo sistema, avrete notato che decine di file e directory sono stati creati senza essere stati elencati in un indice di directory. Oltre a ciò, ogni directory contiene un file chiamato solo con un punto (.) ed un altro con due punti (..) che sono utilizzati in combinazione con il loro numero di inode per determinare la posizione della directory nella struttura ad albero del file system.

In realtà dovreste leggere le pagine Info di **ls**, dal momento che si tratta di un comando molto frequente con molte opzioni utili. Le opzioni si possono combinare, come succede con molti comandi UNIX e le loro opzioni. Una combinazione comune è **ls `-al`**: mostra un lungo elenco di file e le loro proprietà così come le destinazioni a cui punta ciascun collegamento simbolico. **ls `-latr`** mostra gli stessi file, ma ora in ordine di ultima modifica inverso in modo che i file modificati più di recente appaiono alla fine dell'elenco. Ecco qui un paio di esempi:

```
krissie:~/mp3> ls
Albums/   Radio/  Singles/  gene/  index.html

krissie:~/mp3> ls
./   .thumbs  Radio     gene/
../  Albums/  Singles/  index.html

krissie:~/mp3> ls -l Radio/
total 8
drwxr-xr-x    2 krissie krissie  4096 Oct 30  1999 Carolina/
drwxr-xr-x    2 krissie krissie  4096 Sep 24  1999 Slashdot/

krissie:~/mp3> ls -ld Radio/
drwxr-xr-x    4 krissie krissie  4096 Oct 30  1999 Radio/

krissie:~/mp3> ls -ltr
total 20
drwxr-xr-x    4 krissie krissie  4096 Oct 30  1999 Radio/
-rw-r--r--    1 krissie krissie   453 Jan  7  2001 index.html
drwxrwxr-x   30 krissie krissie  4096 Oct 20 17:32 Singles/
drwxr-xr-x    2 krissie krissie  4096 Dec  4 23:22 gene/
drwxrwxr-x   13 krissie krissie  4096 Dec 21 11:40 Albums/
```

In molte versioni Linux **ls** è normalmente un alias di color-ls. Questa caratteristica permette di vedere il tipo dei file senza applicare nessuna opzione a **ls**. Per fare ciò, ogni tipo di file ha un proprio colore. Lo schema standard si trova in `/etc/DIR_COLORS`:

Tabella 3-5. Schema base dei colori in color-ls

Colore	**Tipo di file**
blu	directory
rosso	archivi compressi
bianco	file di testo
rosa	immagini
azzurro	collegamenti
giallo	dispositivi
verde	file eseguibili
rosso lampeggiante	collegamenti interrotti

Maggiori indicazioni si trovano nella pagina man. Nei primi tempi le stesse informazioni [sul tipo di file] venivano rappresentate applicando dei suffissi ad ogni nome di file non standard. Per l'uso in monocromatico (come la stampa di un elenco di directory) e per una leggibilità generale, è ancora utilizzato questo schema:

Tabella 3-6. Schema base dei suffissi per ls

<table>
<tr><th>Carattere</th><th>Tipo di file</th></tr>
<tr><td>nessuno</td><td>file normale</td></tr>
<tr><td>/</td><td>directory</td></tr>
<tr><td>*</td><td>file eseguibile</td></tr>
<tr><td>@</td><td>collegamento</td></tr>
<tr><td>=</td><td>socket</td></tr>
<tr><td>|</td><td>named pipe</td></tr>
</table>

Una descrizione completa delle funzionalità e caratteristiche del comando **ls** può essere letta con **info** ***`coreutils ls`***.

3.3.1.2. Altri strumenti

Per scoprire di più sul tipo dei dati con cui abbiamo a che fare, possiamo usare il comando **file**. Effettuando determinate prove che controllano le proprietà di un file nel file system (numeri magici e test di linguaggio) **file** tenta di fare una educata congettura circa il formato di un file. Alcuni esempi:

```
mike:~> file Documents/
Documents/: directory

mike:~> file high-tech-stats.pdf
high-tech-stats.pdf: PDF document, version 1.2

mike:~> file Nari-288.rm
Nari-288.rm: RealMedia file
```

```
mike:~> file bijlage10.sdw
bijlage10.sdw: Microsoft Office Document

mike:~> file logo.xcf
logo.xcf: GIMP XCF image data, version 0, 150 x 38, RGB Color

mike:~> file cv.txt
CV.txt: ISO-8859 text

mike:~> file image.png
image.png: PNG image data, 616 x 862, 8-bit grayscale, non-interlaced

mike:~> file figure
figure: ASCII text

mike:~> file me+tux.jpg
me+tux.jpg: JPEG image data, JFIF standard 1.01, resolution (DPI),
            "28 Jun 1999", 144 x 144

mike:~> file 42.zip.gz
42.zip.gz: gzip compressed data, deflated, original filename,
          '42.zip', last modified: Thu Nov  1 23:45:39 2001, os: Unix

mike:~> file vi.gif
vi.gif: GIF image data, version 89a, 88 x 31

mike:~> file slide1
slide1: HTML document text

mike:~> file template.xls
template.xls: Microsoft Office Document

mike:~> file abook.ps
abook.ps: PostScript document text conforming at level 2.0

mike:~> file /dev/log
/dev/log: socket

mike:~> file /dev/hda
/dev/hda: block special (3/0)
```

Il comando **file** ha una serie di opzioni, fra cui l'opzione `-z` per curiosare all'interno di file compressi (v. **info** ***file*** per una descrizione dettagliata). Tenete in mente che i risultati di **file** non sono assoluti trattandosi di semplici supposizioni: in altre parole, **file** può essere tratto in inganno.

Perché tutto questo trambusto per i tipi di file e i formati?
In breve, tratteremo di una coppia di strumenti a linea di comando per guardare i *file di puro testo*. Tali strumenti non funzionano se usati con i tipi di file sbagliati. Nel peggiore dei casi, essi manderanno in crash il vostro terminale e/o faranno un sacco di segnali sonori. Se vi capita, basta chiudere la sessione del terminale e ricominciare con una nuova. Ma cercate di evitarlo, perché normalmente è molto fastidioso per le altre persone.

3.3.2. Creare e cancellare file e directory

3.3.2.1. Combinare un pasticcio...

... non è una cosa complicata da fare. Oggi quasi ogni sistema è collegato in rete e

conseguentemente i file vengono copiati da una macchina all'altra. E, specialmente quando si sta lavorando in un ambiente grafico, la creazione di nuovi file è semplicissima e spesso è eseguita senza il consenso dell'utente. Per illustrare il problema, ecco qui il contenuto integrale di una directory di un nuovo utente, creata in un sistema RedHat:

```
[newuser@blob user]$ ls -al
total 32
drwx------   3 user     user          4096 Jan 16 13:32 .
drwxr-xr-x   6 root     root          4096 Jan 16 13:32 ..
-rw-r--r--   1 user     user        24 Jan 16 13:32 .bash_logout
-rw-r--r--   1 user     user       191 Jan 16 13:32 .bash_profile
-rw-r--r--   1 user     user       124 Jan 16 13:32 .bashrc
drwxr-xr-x   3 user     user      4096 Jan 16 13:32 .kde
-rw-r--r--   1 user     user      3511 Jan 16 13:32 .screenrc
-rw-------   1 user     user        61 Jan 16 13:32 .xauthDqztLr
```

Di primo acchito, il contenuto di una directory personale “usata” non sembra neppure in brutte condizioni:

```
olduser:~> ls
app-defaults/ crossover/   Fvwm@    mp3/      OpenOffice.org638/
articles/     Desktop/     GNUstep/ Nautilus/ staroffice6.0/
bin/          Desktop1/    images/  nqc/      training/
brol/         desktoptest/ Machine@ ns_imap/  webstart/
c/            Documents/   mail/    nsmail/   xml/
closed/       Emacs@       Mail/    office52/ Xrootenv.0
```

Ma quando tutti i file e le directory inizianti con un punto sono incluse, esistono 185 oggetti in questa directory. Ciò si spiega perché molte applicazioni hanno le proprie directory e/o file, contenenti impostazioni specifiche dell'utente, nella directory personale di costui. Abitualmente questi file vengono creati la prima volta che avviate l'applicazione. In alcuni casi verrete avvisati quando è necessario creare una nuova directory non ancora esistente, ma il più delle volte tutto viene effettuato automaticamente.

Inoltre nuovi file sono creati apparentemente in continuazione poiché gli utenti vogliono salvare file, tenere versioni differenti dei propri lavori, usare applicazioni Internet e scaricare file e allegati nelle loro macchine locali. Non si ferma. E' evidente che una persona ha bisogno di un preciso schema per avere una panoramica delle cose.

Nella prossima sezione discuteremo su cosa significhi per noi tenere in ordine. Parleremo solo degli strumenti testuali disponibili nella shell, dal momento che gli strumenti grafici sono molto intuitivi e hanno il medesimo aspetto dei gestori di file punta-e-clicca in stile MS Windows, comprese le funzioni grafiche di aiuto ed altre caratteristiche che vi attendete da questo genere di applicazioni. L'elenco seguente è una panoramica dei più popolari gestori di file per GNU/Linux. Molti di loro possono essere avviati dal menu del vostro desktop manager o cliccando sull'icona della vostra directory personale o da linea di comando, dando questi comandi:

- **nautilus**: è il file manager di base in Gnome, il desktop GNU. Eccellente documentazione su come lavorare con questo strumento può essere trovata su http://www.gnome.org.
- **konqueror**: il gestore di file usato abitualmente nel desktop KDE. Il manuale si trova su http://docs.kde.org.

- **mc**: Midnight Commander, il file manager di Unix con lo stile del Norton Commander. Tutta la documentazione disponibile su http://gnu.org/directory o su un mirror come http://www.ibiblio.org.

Queste applicazioni meritano certamente di essere provate e solitamente impressionano i nuovi arrivati di Linux, anche solo perché esiste una così grande varietà: questi sono solo gli strumenti più diffusi per gestire directory e file, mentre molti altri progetti sono in corso di sviluppo. Ora scopriamone il funzionamento e vediamo come questi strumenti grafici utilizzano i comuni comandi UNIX.

3.3.2.2. Gli strumenti

3.3.2.2.1. Creare directory

Una maniera per tenere le cose in ordine è quella di assegnare a certi file locazioni specifiche di base creando directory e sottodirectory (o cartelle e sottocartelle se preferite). Ciò si fa con il comando **mkdir**:

```
richard:~> mkdir archive

richard:~> ls -ld archive
drwxrwxrwx  2 richard richard       4096 Jan 13 14:09 archive/
```

La creazione di directory e sottodirectory in una sola mossa si fa usando l'opzione `-p`:

```
richard:~> cd archive

richard:~/archive> mkdir 1999 2000 2001

richard:~/archive> ls
1999/  2000/  2001/

richard:~/archive> mkdir 2001/reports/Restaurants-Michelin/
mkdir: cannot create directory '2001/reports/Restaurants-Michelin/':
No such file or directory

richard:~/archive> mkdir -p 2001/reports/Restaurants-Michelin/

richard:~/archive> ls 2001/reports/
Restaurants-Michelin/
```

Se il nuovo file ha bisogno di ulteriori permessi oltre a quelli di base per la creazione, i nuovi diritti di accesso possono essere impostati in una sola mossa, usando ancora il comando **mkdir** (v. pagine Info per maggiori informazioni). Stiamo per trattare le modalità d'accesso nella prossima sezione dedicata alla sicurezza dei file.

Il nome di una directory deve attenersi alle stesse regole applicate ai nomi dei comuni file. Una delle restrizioni più importanti è che non ci possono esistere due file con lo stesso nome in una directory (ma ricordate che Linux è un sistema operativo che distingue tra lettere maiuscole e minuscole [*case sensitive*] come UNIX). Virtualmente non esistono limiti alla lunghezza del nome di un file, ma di solito si tiene più corto di 80 caratteri in modo da farlo stare in una sola linea di terminale. Potete usare qualsiasi carattere a piacere nel nome di un file, sebbene sia consigliabile escludere i caratteri che hanno un significato speciale per la shell. In caso di dubbi, consultate

l'Appendice C.

3.3.2.2.2. Spostare file

Ora che abbiamo strutturato correttamente la nostra directory personale, è tempo di riorganizzare i file non classificati usando il comando **mv**:

```
richard:/archive> mv ../report[1-4].doc reports/Restaurants-Michelin/
```

Questo comando serve anche per rinominare i file:

```
richard:~> ls To_Do
-rw-rw-r--    1 richard richard     2534 Jan 15 12:39 To_Do

richard:~> mv To_Do done

richard:~> ls -l done
-rw-rw-r--    1 richard richard     2534 Jan 15 12:39 done
```

E' evidente che cambia solo il nome del file: tutte le altre proprietà rimangono uguali.

Dettagliate informazioni sulla sintassi e le caratteristiche del comando **mv** si possono trovare nelle pagine man o Info. L'uso di tale documentazione dovrebbe essere il vostro primo pensiero quando incontrate un problema: facilmente la risposta ad esso si trova proprio nella documentazione di sistema. Come gli utenti esperti consultano le pagine man quotidianamente, così i principianti dovrebbero leggerle ogni volta. Dopo un po' imparerete le opzioni più comuni dei normali comandi, ma avrete ancora bisogno della documentazione come fonte primaria di informazioni. Notate che le informazioni contenute negli HOWTO, FAQ, pagine man e così via, stanno lentamente trasfondendosi nelle pagine Info, che oggi sono la fonte più aggiornata di documentazione in linea (cioè disponibile per la lettura sul sistema).

3.3.2.2.3. Copiare file

La copia di file e directory si effettua con il comando **cp**. Una funzione utile è la copia ricorsiva (copia di tutti i file sottostanti e delle sottodirectory) che si ottiene aggiungendo l'opzione `-R` a **cp**. La sintassi generale è

cp [-R] dal_file al_file

Come esempio ecco il caso dell'utente *newguy* che desidera le stesse impostazioni del desktop Gnome che ha l'utente *oldguy*. Un modo per risolvere il problema è copiare le impostazioni di *oldguy* nella directory personale di *newguy*:

```
victor:~> cp -R ../oldguy/.gnome/ .
```

Ciò restituirà alcuni errori legati ai permessi sui file, ma tutti questi errori hanno a che fare con i file privati di cui, in ogni caso, *newguy* non ha necessità. Tratteremo nella prossima parte come cambiare questi permessi in caso essi costituiscano veramente un problema.

3.3.2.2.4. Rimuovere i file

Utilizzate il comando **rm** per rimuovere singoli file, **rmdir** per rimuovere directory vuote (usate **ls -a** per verificare se una directory è vuota o meno). Il comando **rm** ha anche opzioni per rimuovere directory non vuote con tutte le loro sottodirectory (leggete le pagine Info per queste opzioni piuttosto pericolose).

Quanto vuota può essere una directory?
E' normale che le directory . (punto) e .. (punto-punto) non possano essere rimosse, dal momento che esse sono necessarie anche in una directory vuota per stabilire il livello delle directory nella gerarchia del file system.

In Linux, esattamente come in UNIX, non esiste un cestino - almeno non nella shell, sebbene ci siano numerose soluzioni per la modalità grafica. Cosicché un file, una volta rimosso, non esiste più ed in genere non c'è modo di recuperarlo a meno che non abbiate dei backup o siate veramente rapidi e abbiate un amministratore di sistema realmente bravo. Per proteggere il principiante da questo inconveniente, è possibile attivare con l'opzione `-i` il comportamento interattivo dei comandi **rm**, **cp** e **mv**. In tal caso il sistema non agirà immediatamente su richiesta ma, invece, chiederà conferma, richiedendo un clic aggiuntivo sul tasto **Invio** per produrre il danno:

```
mary:~> rm -ri archive/
rm: descend into directory 'archive'? y
rm: descend into directory 'archive/reports'? y
rm: remove directory 'archive/reports'? y
rm: descend into directory 'archive/backup'? y
rm: remove directory 'archive/backup/sysbup200112.tar'? y
rm: remove directory 'archive/backup'? y
rm: remove directory 'archive'? y
```

Tratteremo di come rendere automatica tale opzione nel Capitolo 7 in cui si parla della personalizzazione del vostro ambiente di shell.

3.3.3. Trovare i file

3.3.3.1. Utilizzo delle caratteristiche della shell

Abbiamo già visto nell'esempio dello spostamento dei file come la shell può gestire più file alla volta. In quell'esempio la shell rileva automaticamente ciò che l'utente vuol dire in base alle richieste contenute tra le parentesi quadre “[“ e “]”. La shell può sostituire serie di numeri e di lettere maiuscole o minuscole indifferentemente. Essa sostituisce pure quanti caratteri volete con un asterisco ed un solo carattere con un punto di domanda.

Tutti i tipi di sostituzioni possono essere usati contemporaneamente: la shell è molto logica in ciò. La shell Bash, per esempio, non ha problemi con espressioni come **ls** ***`dirname/*/*/*[2-3]`***.

In altre shell, l'asterisco è comunemente usato per ridurre gli sforzi di battitura: una persona potrebbe scrivere **cd** ***`dir*`*** al posto di **cd** ***`directory`***. In Bash, comunque, ciò non è necessario poiché la shell GNU ha una funzione chiamata completamento dei nomi dei file: significa che voi

potete battere i primi caratteri di un comando (ovunque) o di un file (nella directory corrente) e, se non c'è possibilità di confusione, la shell indovinerà cosa cosa intendete dire. Per esempio, in una directory contenente molti file, potete controllare se c'è qualche file che inizia con la lettera A battendo soltanto **ls *A*** e premendo il tasto **Tab** per due volte, piuttosto che premere **Invio**. Se esiste un solo file che inizia per "A", questo verrà mostrato come argomento di **ls** (o qualsiasi altro comando di shell) immediatamente.

3.3.3.2. which

Un modo piuttosto semplice per trovare i file è ricorrere al comando **which**, che, per l'appunto, ricerca i file richiesti nelle directory elencate nel percorso di ricerca dell'utente. Naturalmente, dal momento che il percorso di ricerca contiene solo percorsi a directory con programmi eseguibili, **which** non funziona con i file normali. Il comando **which** è utile quando appaiono messaggi tipo "Command not found". Nell'esempio qui sotto, l'utente *tina* non può usare il programma **acroread**, mentre il suo collega non incontra alcun problema sullo stesso sistema. Il problema è simile a quello del PATH (o PERCORSO) della parte precedente: il collega di Tina le dice che lui può vedere il programma richiesto in `/opt/acroread/bin`, ma questa directory non si trova nel percorso di Tina:

```
tina:~> which acroread
/usr/bin/which: no acroread in (/bin:/usr/bin:/usr/bin/X11)
```

Il problema può essere risolto scrivendo il percorso completo del comando da avviare o riesportando il contenuto della variabile `PATH`:

```
tina:~> export PATH=$PATH:/opt/acroread/bin

tina:~> echo $PATH
/usr:/usr/bin:/usr/bin/X11:/opt/acroread/bin
```

Con **which** si può anche controllare se un comando è un alias di un altro:

```
gerrit:~> which -a ls
ls is aliased to 'ls -F --color=auto'
ls is /bin/ls
```

Se ciò non dovesse funzionare, utilizzate il comando **alias**:

```
tille@www:~/mail$ alias ls
alias ls='ls --color'
```

3.3.3.3. find e locate

Questi sono i veri strumenti utilizzati per trovare altri percorsi oltre a quelli elencati nel percorso di ricerca [*search path*]. Lo strumento **find**, noto sin da UNIX, è molto potente, cosa che può determinare una certa difficoltà di sintassi. Tale comando non solo vi consente di ricercare nomi di file, ma accetta pure le dimensioni dei file, la data dell'ultima modifica e altre proprietà dei file quali criteri di ricerca. L'uso più comune è la ricerca dei nomi dei file:

find <percorso> -name <*stringadiricerca*>

Ciò può essere così interpretato: “Cerca tutti i file e le sottodirectory contenute in un dato percorso e stampa i nomi dei file contenenti la stringa di ricerca nel loro nome (non nel loro contenuto)”.

Altra applicazione di **find** si ha per ricercare file di una certa misura, come nell'esempio seguente, dove l'utente *peter* vuole trovare tutti i file della corrente directory o di una delle sue sottodirectory, che sono più grandi di 5 MB:

```
peter:~> find . -size +5000k
psychotic_chaos.mp3
```

Se scavate nelle pagine man, vedrete che **find** può compiere operazioni sui file trovati. Un semplice esempio è la rimozione dei file. E' meglio prima provare senza l'opzione `-exec` che siano stati selezionati i file giusti e poi ridare il comando per cancellarli. Sotto, cerchiamo file che finiscono per `.tmp`:

```
peter:~> find . -name “*.tmp” -exec rm {} \;

peter:~>
```

Ottimizzate!
Tale comando richiamerà **rm** tante volte quante sarà rintracciato un file corrispondente ai criteri di ricerca. Nel peggior caso potrebbe essere migliaia o milioni di volte. E' abbastanza un bel carico sul vostro sistema.
Un modo di procedere più realistico sarebbe quello di usare una pipe (|) e lo strumento **xargs** con **rm** per argomento. In questa maniera il comando **rm** è chiamato solo quando la linea di comando è piena, al posto di ogni file. Guardate il Capitolo 5 per maggiori informazioni sull'uso della redirezione dell'I/O per facilitare i compiti di ogni giorno.

Successivamente (nel 1999 secondo le pagine man, dopo 20 anni da **find**) è stato sviluppato **locate**. Questo programma è più facile da usare, ma più limitato rispetto a **find**, dal momento che il suo output si basa su un database indicizzato dei file che viene aggiornato solo una volta al giorno. D'altra parte una ricerca nel database **locate** richiede minori risorse rispetto a **find** e quindi mostra i risultati quasi istantaneamente.

Molte distribuzioni Linux oggi usano **slocate**, un locate migliorato dal punto di vista della sicurezza, la versione moderna di **locate** che impedisce agli utenti di ottenere informazioni in uscita che essi non hanno il diritto di leggere. I file nella directory personale di *root* sono un esempio: questi non sono normalmente accessibili al pubblico. Un utente che desideri rintracciare qualcuno che conosca la C shell può dare il comando **locate `.cshrc`** per vedere tutti gli utenti che hanno un file personalizzato di configurazione della C shell. Supponendo che gli utenti *root* e *jenny* stiano facendo girare una C shell, solo il file `/home/jenny/.cshrc` verrà mostrato e non quello nella home directory di *root*. In molti sistemi **locate** è un collegamento simbolico al programma **slocate**:

```
billy:~> ls -l /usr/bin/locate
lrwxrwxrwx 1 root slocate  7 Oct 28 14:18 /usr/bin/locate -> slocate*
```

L'utente *tina* avrebbe potuto usare **locate** per trovare l'applicazione desiderata:

```
tina:~> locate acroread
/usr/share/icons/hicolor/16x16/apps/acroread.png
/usr/share/icons/hicolor/32x32/apps/acroread.png
/usr/share/icons/locolor/16x16/apps/acroread.png
/usr/share/icons/locolor/32x32/apps/acroread.png
/usr/local/bin/acroread
/usr/local/Acrobat4/Reader/intellinux/bin/acroread
/usr/local/Acrobat4/bin/acroread
```

Le directory che non contengono il nome `bin` non possono contenere il programma (esse non contengono file eseguibili). Restano tre possibilità. Il file in `/usr/local/bin` è il solo che *tina* avrebbe ricercato: è un collegamento allo script di shell che avvia l'attuale programma:

```
tina:~> file /usr/local/bin/acroread
/usr/local/bin/acroread: symbolic link to ../Acrobat4/bin/acroread

tina:~> file /usr/local/Acrobat4/bin/acroread
/usr/local/Acrobat4/bin/acroread: Bourne shell script text executable

tina:~> file /usr/local/Acrobat4/Reader/intellinux/bin/acroread
/usr/local/Acrobat4/Reader/intellinux/bin/acroread: ELF 32-bit LSB
executable, Intel 80386, version 1, dynamically linked (uses
shared libs), not stripped
```

Per mantenere il percorso più breve possibile, in modo tale che il sistema non debba cercare troppo a lungo ogni volta che un utente vuole eseguire un programma, aggiungeremo `/usr/local/bin` al percorso e non le altre directory, contenti solamente i file binari di uno specifico programma, mentre `/usr/local/bin` contiene altri utili programmi.

Di nuovo: una descrizione delle funzionalità complete di **find** e di **locate** possono essere trovate nelle pagine Info.

3.3.3.4. Il comando grep

3.3.3.4.1. Filtraggio delle linee in generale

Un semplice ma potente programma, **grep**, viene utilizzato per filtrare le linee in ingresso e restituire certi modelli in uscita. Esistono letteralmente migliaia di applicazioni per il programma **grep**. Nell'esempio seguente, *jerry* usa **grep** per vedere come ha utilizzato **find**:

```
jerry:~> grep -a find .bash_history
find . -name userinfo
man find
find ../ -name common.cfg
```

Ricerca nella history

Pure utile in questi casi è la funzione di ricerca di **bash**, che si attiva premendo insieme **Ctrl+R**, come nell'esempio dove vogliamo controllare nuovamente che cosa abbiamo fatto con l'ultimo **find**:

```
thomas ~> ^R
(reverse-i-search)'find': find '/home/thomas' -name *.xml
```

Battete la vostra stringa di ricerca al prompt di ricerca. Più caratteri batterete e più ridurrete i risultati della ricerca. Questa legge la cronologia (*history*) del comando della sessione di shell (che viene scritta in `.bash_history` della vostra directory personale quando voi chiudete quella sessione). Viene mostrata la più recente apparizione della vostra stringa di ricerca. Se volete vedere i comandi precedenti che contengono la medesima stringa, battete di nuovo **Ctrl+R** (v. pagine Info su Bash per saperne di più).

Tutti gli Unix con un minimo di decenza hanno un dizionario in linea: così pure Linux. Il dizionario è un elenco delle parole note in un file chiamato `words`, posizionato in `/usr/share/dict`. Per controllare rapidamente la scrittura corretta di una parola, non serve un'applicazione grafica:

```
william:~> grep pinguin /us/share/dict/words

william:~> grep penguin /us/share/dict/words
penguin
penguins
```

Dizionario contro elenco di parole

Alcune distribuzioni offrono il comando **dict** che mette a disposizione maggiori funzionalità rispetto alla semplice ricerca di parole di un elenco.

Chi è il proprietario della home directory dopo la mia? Hey, c'è il suo numero telefonico!

```
lisa:~> grep gdbruyne /etc/passwd
gdbruyne:x:981:981:Guy Debruyne, tel 203234:/gdbruyne:/bin/bash
```

E, di nuovo, qual è l'indirizzo e-mail di Arno?

```
serge:~/mail> grep -i arno *
sent-mail: To: <Arno.Hintjens@celeb.com>
sent-mail: On Mon, 24 Dec 2001, Arno Hintjens@celeb.com wrote:
```

find e **locate** vengono spesso usati in combinazione con **grep** per realizzare delle ricerche serie. Per maggiori informazioni, v. Capitolo 5 sulla redirezione dell'I/O.

3.3.3.4.2. Caratteri speciali

I caratteri che hanno uno speciale significato per la shell devono essere segnalati con il carattere di *escape*: nella Bash è la sbarra inversa, come in molte shell, e determina lo speciale significato del carattere successivo. La shell conosce svariati caratteri speciali tra cui, i più comuni, "/",".","?" e "*". Un elenco completo si può trovare nelle pagine Info e nella documentazione della vostra shell.

Per esempio, dire che volete vedere il file "*" al posto di tutti i file di una directory, dovete usare

less*

Lo stesso vale per i nomi di file contenenti uno spazio:

cat `This\ File`

3.3.4. Più modi di vedere il contenuto dei file

3.3.4.1. In generale

Eccetto **cat**, che non fa molto di più di inviare i file allo standard output, ci sono altri strumenti per vedere il contenuto dei file.

Naturalmente la maniera più semplice sarebbe quella di usare strumenti grafici al posto di quelli a linea di comando. Nell'introduzione abbiamo già dato un rapido sguardo ad una applicazione da ufficio, OpenOffice.org. Altri esempi sono GIMP (si avvia con **gimp** dalla linea di comando) GNU Image Manipulating Program, **xpdf** per vedere i file PDF (Portable Document Format), GhostView (**gv**) per vedere i file PostScript, Mozilla/Firefox, **links** (un browser in modalità testo), Konqueror, Opera e molti altri per quanto riguarda i contenuti del web, XMMS, Cdplay e altri per i contenuti dei file multimediali, AbiWord, Gnumeric, Koffice, ecc..., per tutti i generi di applicazioni d'ufficio e così via. Esistono migliaia di applicazioni Linux: elencarle tutte richiederebbe giorni.

Invece noi ci concentreremo sulle applicazioni di shell o in modalità testo, che sono la base di tutte le altre applicazioni. Questi comandi lavorano meglio in un ambiente testuale su file contenenti testo. In caso di incertezza, controllate prima con il comando **file**.
Vediamo così quali strumenti testuali utilizzabili per curiosare all'interno dei file sono a nostra disposizione.

Problemi di font
Gli strumenti di puro testo come quelli che ora tratteremo, spesso incontrano problemi con i file di ”puro” testo a causa della codifica dei font utilizzata con quei file. Caratteri speciali, tipo i caratteri alfabetici accentati, i caratteri cinesi e gli altri caratteri provenienti da linguaggi che usano insiemi di caratteri differenti dalla codifica base *en_US* e così via, sono poi visualizzati in modo errato o rimpiazzati da robaccia illeggibile. Questi problemi saranno discussi nella Sezione 7.4.

3.3.4.2. “less is more”

Senza dubbio avrete sentito prima o poi qualcuno pronunciare questa frase lavorando in un ambiente UNIX. Un pezzetto di storia di UNIX spiega ciò:

- Prima ci fu **cat**. L'uscita dei dati fluiva in modo incontrollabile.
- Poi venne **pg**, che si può ancora trovare in vecchi UNIX. Questo comando produce il testo in uscita una pagina alla volta.
- Il programma **more** fu una versione rivista di **pg**. Questo comando è ancora disponibile in ogni sistema Linux.
- **less** è la versione GNU di **more** ed ha delle caratteristiche extra permettendo l'evidenziazione delle stringhe di ricerca, lo scorrimento all'indietro, ecc... La sintassi è molto semplice:

less `nome_del_file`

Maggiori informazioni si trovano nelle pagine Info.

Conoscete già gli “impaginatori” (pager) dal momento che essi sono stati usati per vedere le pagine man.

3.3.4.3. I comandi head e tail

Questi due comandi mostrano rispettivamente le *n* prime/ultime linee di un file. Per osservare gli ultimi dieci comandi usati:

```
tony:~> tail -10 .bash_history
locate configure | grep bin
man bash
cd
xavtv &
grep usable /usr/share/dict/words
grep advisable /usr/share/dict/words
info quota
man quota
echo $PATH
frm
```

head funziona in modo simile. Il comando **tail** ha una utile funzione di mostrare in continuazione le ultime n linee di un file che cambia costantemente. Tale opzione `-f` viene spesso usata dagli amministratori di sistema per controllare i file di log (maggiori informazioni si trovano nei file di documentazione del sistema).

3.3.5. Collegare i file

3.3.5.1. Tipi di collegamento

Dal momento che ora sappiamo di più sui file e la loro rappresentazione nel file system, è un gioco da ragazzi capire i collegamenti (*link, shortcut* o *scorciatoie*). Un link non è niente di più di un modo per assegnare due o più nomi di file allo stesso insieme di dati di file. Esistono due maniere per ottenere ciò:

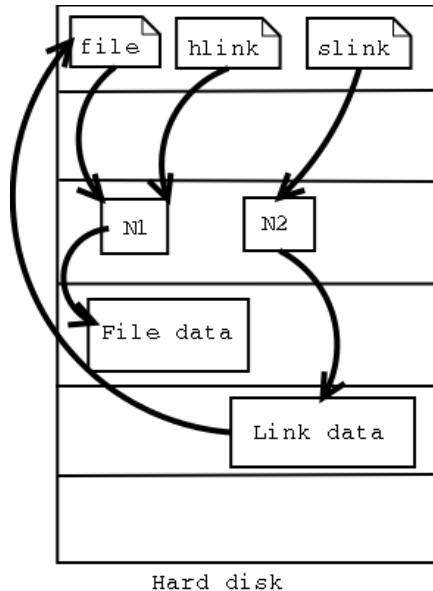
- Hard link (*collegamento fisso*): si associano due o più nomi di file al medesimo inode. Gli hard link condividono gli stessi blocchi di dati nel disco rigido, mentre continuano a comportarsi come file indipendenti.

  C'è uno svantaggio immediato: i collegamenti fissi non possono essere creati tra partizioni in quanto i numeri di inode sono unici per una data partizione.

- Soft link o link simbolico (*collegamento simbolico* o, in breve, *symlink*): un piccolo file che punta ad un altro file. Un collegamento simbolico contiene il percorso del file di destinazione (*target file*) invece di una locazione fisica sul disco rigido.

Siccome con questo metodo non si fa uso di inode, i collegamenti possono espandersi tra le partizioni.

I due tipi di collegamento si comportano in modo simile, ma non sono la stessa cosa, come viene illustrato nello schema seguente:

Figura 3-2. Il meccanismo dei collegamenti hard e soft

Notate che la rimozione del target file di un link simbolico rende inutile quest'ultimo.

Ogni comune file è inizialmente un hardlink. I collegamenti fissi non possono estendersi a più directory del momento che essi fanno riferimento agli inode e i numeri di inode sono unici per una data partizione.

Si potrebbe arguire che ci sia un terzo tipo di collegamento, il link in spazio utente (*user-space link*), che sarebbe simile alle scorciatoie di MS Windows. Questi sono dei file contenenti metadati che possono essere interpretati solo dal gestore grafico di file. Per il kernel e la shell questi sono solo normali file. Essi potrebbero terminare con i sufissi *.desktop* o *.lnk*: possiamo trovare un esempio in `~/.gnome-desktop`:

```
[dupont@boulot .gnome-desktop]$ cat La\Maison\ Dupont
[Desktop Entry]
Encoding=Legacy-Mixed
Name=La Maison Dupont
Type=X-nautilus-home
X-Naautilus-Icon=temp-home
URL=file:///home/dupont
```

Questo esempio è tratto da un desktop KDE:

```
[lena@venus Desktop]$ cat camera
[Desktop Entry]
Dev=/dev/sda1
```

```
FSType=auto
Icon=memory
MountPoint=/mnt/camera
Type=FSDevice
X-KDE-Dynamic-Device=true
```

La creazione di questo tipo di collegamento è abbastanza facile ricorrendo alle funzioni del vostro ambiente grafico. Se aveste bisogno di aiuto, la vostra documentazione di sistema potrebbe essere la vostra prima risorsa.

Nella prossima sezione studieremo la creazione di link simbolici in stile UNIX ricorrendo alla linea di comando.

3.3.5.2. Creare collegamenti simbolici

Il link simbolico (symlink) è particolarmente interessante per gli utenti principianti: esso è abbastanza facile da riconoscere e non dovete preoccuparvi delle partizioni.

Il comando per creare link è **ln**. Per creare symlink dovete usare l'opzione `-s`:

ln `-s filedacollegare nomedelcollegamento`

Nell'esempio seguente, l'utente *freddy* crea un link in una sottodirectory di quella personale ad una directory di un'altra parte del sistema:

```
freddy:~/music> ln -s /opt/mp3/Queen/ Queen

freddy:~/music> ls -l
lrwxrwxrwx  1 freddy  freddy  17 Jan 22 11:07 Queen-> /opt/mp3/Queen
```

I link simbolici sono sempre file molto corti, mentre gli hard link hanno le stesse dimensioni del file originale.

L'uso dei link simbolici è ampiamente diffuso. Essi sono spesso usati per risparmiare spazio sul disco, per fare una copia di un file al fine di soddisfare le richieste di installazione di un nuovo programma che si attende di trovare in un altro luogo: sono utilizzati per correggere script che improvvisamente devono girare in un nuovo ambiente e possono generalmente risparmiare un sacco di lavoro. Un amministratore di sistema può decidere di spostare le directory personali degli utenti in un altro posto, `disk2` per esempio, ma se vuole che tutto funzioni come prima, come il file `/etc/passwd`, con uno sforzo minimo può creare un symlink da `/home` alla nuova posizione `/disk2/home`.

3.4. La sicurezza dei file

3.4.1. Diritti di accesso: la prima linea di difesa di Linux

Il modello di sicurezza di Linux si basa su quello utilizzato nei sistemi UNIX, ed è inflessibile come quest'ultimo (e qualche volta di più), che è già abbastanza robusto. In un sistema Linux ogni file è

di proprietà di un utente e di un gruppo di utenti. Esiste anche una terza categoria di utenti che non sono né l'utente proprietario, né appartengono al gruppo proprietario del file. I permessi di lettura, scrittura ed esecuzione possono essere garantiti o negati.

Abbiamo già usato l'opzione *long* per elencare i file con il comando **ls -l**, sebbene per altri motivi. Tale comando mostra anche i permessi dei file attribuiti alle tre categorie di utenti: essi vengono indicati da nove caratteri che seguono il primo carattere, costituito dall'indicatore del tipo di file posto all'inizio della linea delle proprietà del file. Come vedete nell'esempio seguente, le prime tre lettere di questa serie di nove mostra i diritti di accesso dell'attuale utente proprietario del file. Le successive tre lettere sono relative al gruppo a cui appartiene il file, mentre le ultime tre riguardano gli altri utenti. I permessi sono sempre nello stesso ordine: lettura, scrittura ed esecuzione per l'utente *(user),* il gruppo (*group*) e gli altri (*other*). Alcuni esempi:

```
marise:~> ls -l To_Do
-rw-rw-r--    1 marise  users         5 Jan 15 12:39 To_Do
marise:~> ls -l /bin/ls
-rwxr-xr-x    1 root    root      45948 Aug  9 15:01 /bin/ls*
```

Il primo è un normale file (primo trattino). Gli utenti con nome *marise* o appartenenti al gruppo *users* possono leggere e scrivere (modificare/spostare/cancellare) il file ma non possono metterlo in esecuzione (secondo e terzo trattino). Tutti gli altri utenti hanno solo il permesso di leggere questo file, ma non possono né scriverlo, né eseguirlo (quarto e quinto trattino).

Il secondo esempio è un file eseguibile. La differenza: ognuno può eseguire questo programma ma dovete essere *root* per poterlo modificare.

Le pagine Info spiegano dettagliatamente come il comando **ls** gestisce la rappresentazione dei diritti di accesso (v. la sezione *What information is listed*).

Per un facile utilizzo con i comandi, sia i diritti o modi di accesso sia i gruppi hanno un codice: osservate le tabelle seguenti.

Tabella 3-7. I codici delle modalità di accesso

Codice	Significato
0 o -	Il diritto di accesso che si pensa essere in questa posizione non viene concesso
4 o r	L'accesso in lettura è concesso alla categoria di utenti definita in questa posizione
2 o w	Il permesso di scrittura è concesso alla categoria di utenti definita in questa posizione
1 o x	Il permesso di esecuzione è concesso alla categoria di utenti definita in questa posizione

Tabella 3-8. Codici dei gruppi d'utenti

Codici	Significato
u	permessi dell'utente (*user*)
g	permessi del gruppo (*group*)
o	permessi per gli altri (*others*)

Questo chiaro schema viene applicato molto rigidamente, cosa che rende possibile un elevato livello di sicurezza perfino senza la sicurezza di rete. Fra le altre funzioni, lo schema di sicurezza si occupa dell'accesso ai programmi da parte degli utenti, può fornire file in base alle necessità di conoscenza e protegge i dati sensibili come le directory personali e i file di configurazione del sistema.

Dovreste sapere che cosa è il vostro nome utente. In caso contrario, può essere mostrato usando il comando **id**, che mostra anche il gruppo base a cui appartenete ed eventualmente gli altri gruppi di cui siete membri:

```
tilly:~> id
uid=504(tilly) gid=504(tilly) groups=504(tilly),100(users),2051(org)
```

Il vostro nome utente è conservato anche nella variabile d'ambiente `USER`:

```
tilly:~> echo $USER
tilly
```

3.4.2. Gli strumenti

3.4.2.1. Il comando chmod

Una normale conseguenza dell'applicazione di rigidi permessi dei file (e qualche volta anche una seccatura) è che i diritti d'accesso si dovranno cambiare per ogni genere di motivo. Per fare ciò usiamo il comando **chmod** ed alla fine *to chmod* diventa un verbo inglese quasi accettabile, significando il cambio delle modalità d'accesso ad un file. Il comando **chmod** può essere utilizzato con opzioni alfanumeriche o numeriche come meglio ritenete.

L'esempio seguente usa le opzioni alfanumeriche per risolvere un problema che di solito si presenta con i nuovi utenti:

```
asim:~> ./hello
bash: ./hello: bad interpreter: Permission denied

asim:~> cat hello
#!/bin/bash
echo “Hello, World”

asim:~> ls -l hello
-rw-rw-r--    1 asim    asim    32 Jan 15 16:29 hello

asim:~> chmod u+x hello

asim:~> ./hello
Hello, World

asim:~> ls -l hello
-rwxrw-r--    1 asim    asim    32 Jan 15 16:29 hello*
```

Gli operatori + e - sono usati per concedere o negare un certo diritto ad un certo gruppo: sono permesse combinazioni separate. Le pagine Info e man contengono degli utili esempi. Eccone qui

un altro che trasforma il file del precedente esempio in un file personale dell'utente *asim*:

```
asim:~> chmod u+rwx,go-rwx hello

asim:~> ls-l hello
-rwx------    1 asim    asim    32 Jan 15 16:29 hello*
```

Il tipo di problema manifestantesi in un messaggio di errore che dice che il permesso è negato da qualche parte, normalmente dipende in molti casi da un problema con i privilegi d'accesso. Anche commenti come “Ieri funzionava” o “Funziona quando lo avvio come root” derivano il più delle volte da errati permessi dei file.

Usando **chmod** con argomenti numerici, i valori di ogni diritto d'accesso consentito devono essere conteggiati insieme per gruppo. Così otteniamo un numero di tre cifre che è il valore simbolico delle impostazioni create con **chmod**. La tabella seguente elenca le combinazioni più comuni:

Tabella 3-9. Protezione dei file con chmod

Comando	Significato
chmod ***400*** **file**	Per proteggere un file contro sovrascritture accidentali.
chmod ***500 directory***	Per proteggervi da cancellazioni, ridenominazioni o spostamenti accidentali di file da questa directory.
chmod ***600*** **file**	Un file privato che può essere modificato solo dall'utente che ha dato questo comando.
chmod ***644*** **file**	Un file leggibile pubblicamente che può essere modificato solo dall'utente che l'ha messo a disposizione.
chmod ***660*** **file**	Gli utenti appartenenti al vostro gruppo possono modificare questo file mentre gli altri non possono assolutamente accedervi.
chmod ***700*** **file**	Protegge un file contro ogni accesso da parte degli altri utenti, mentre l'utente proprietario ha ancora il pieno accesso.
chmod ***755*** **directory**	Per file che dovrebbero essere leggibili ed eseguibili dagli altri, ma modificabili solo dall'utente proprietario.
chmod ***775*** **file**	Modalità standard per la condivisione di un file in un gruppo.
chmod ***777*** **file**	Chiunque può fare di tutto su questo file.

Se inserite un numero con meno di tre cifre come argomento di **chmod**, i caratteri omessi vengono rimpiazzati con degli zero a cominciare da sinistra. Effettivamente c'è una quarta cifra nei sistemi Linux, che precede le prime tre ed imposta speciali modalità di accesso. Qualsiasi informazione su ciò (e molto altro ancora) si trova nelle pagine Info.

3.4.2.2. Accesso ad un altro gruppo

Quando battete **id** nella linea di comando, ottenete un elenco di tutti i gruppi a cui potete

appartenere, preceduti dai vostri nome utente e ID e dal nome del gruppo e ID con cui siete correntemente connessi. Comunque in molti sistemi Linux potete essere attivamente collegati solo con un gruppo alla volta. Di base questo *gruppo attivo* o *primario* è quello assegnatovi nel file `/etc/passwd`. Il quarto campo di questo file contiene l'ID del gruppo primario degli utenti, che si può ricercare nel file `/etc/group`. Un esempio:

```
asim:~> id
uid=501(asim) gid=501(asim) groups=100(users),501(asim),3400(web)

asim:~> grep asim /etc/passwd
asim:x:501:501:Asim El Baraka:/home/asim:/bin/bash

asim:~> grep 501 /etc/group
asim:x:501:
```

Il quarto campo nella linea da `/etc/passwd` contiene il valore “501” che rappresenta il gruppo *asim* nell'esempio precedente. Da `/etc/group` possiamo ottenere il nome corrispondente a questo ID del gruppo. Quando si connette al sistema, questo è il gruppo a cui appartiene *asim*.

Schema del gruppo privato dell'utente

Per consentire una maggiore flessibilità, molti sistemi Linux seguono il cosiddetto *user private group scheme* (o schema del gruppo privato dell'utente) che assegna ciascun utente in primo luogo al proprio gruppo. Tale gruppo è un gruppo che contiene solo questo particolare utente, da qui il nome “gruppo privato”. Normalmente questo gruppo ha lo stesso nome del nome di login dell'utente, cosa che può creare una leggera confusione.

A parte il proprio gruppo privato, l'utente *asim* può anche stare nei gruppi *users* e *web*. Poiché questi sono gruppi secondari per tale utente, costui dovrà usare **newgrp** per registrarsi in uno di questi gruppi (utilizzate **gpasswd** per impostare prima la password del gruppo). Nell'esempio, *asim* deve creare file di proprietà del gruppo *web*.

```
asim:/var/www/html> newgrp web

asim:/var/www/html> id
uid=501(asim) gid=3400(web) groups=100(users),501(asim),3400(web)
```

Ora, quando *asim* creerà nuovi file, essi saranno di proprietà del gruppo *web* al posto del gruppo *asim*:

```
asim:/var/www/html> touch test

asim:/var/www/html> ls -l test
-rw-rw-r--  1 asim web   0 Jun 10 15:38 test
```

Registrarsi in un nuovo gruppo vi permette di evitare l'uso di **chown** (v. Sezione 3.4.2.4) o di contattare il vostro amministratore di sistema per cambiare le proprietà per voi.

Per maggiori informazioni date un'occhiata alle pagine man riguardanti **newgrp**.

3.4.2.3. La “maschera” dei file

Quando si salva un file da qualche parte, questo per prima cosa viene sottoposto alla procedura standard di sicurezza. I file senza permessi non esistono in Linux. I permessi dei file standard sono stabiliti dalla *mask* (maschera) per la creazione di nuovi file. Il valore di questa maschera può essere visualizzato mediante l'uso del comando **umask**:

```
bert:~> umask
0002
```

Invece di aggiungere i valori simbolici a ciascuno, come con **chmod**, per calcolare i permessi su un nuovo file questi devono essere sottratti dai diritti d'accesso totali possibili. Nell'esempio precedente, comunque, vediamo 4 valori pur essendoci solo 3 categorie di permessi: *user*, *group* e *other*. Il primo zero fa parte delle impostazioni speciali degli attributi dei file che tratteremo nelle Sezioni 3.4.2.4 e 4.1.6. Potrebbe succedere che nel vostro sistema non venga neppure mostrato il primo zero quando battete il comando **umask** e che vediate solo i tre numeri che rappresentano la maschera di base per la creazione dei file.

Ogni sistema derivato da UNIX ha una funzione di sistema per la creazione di nuovi file, funzione che viene chiamata ogni qualvolta un utente utilizza un programma che crea nuovi file, per esempio, quando si scarica un file da internet, quando si salva un nuovo documento di testo e così via. Questa funzione crea sia file che directory nuovi. Il permesso totale di leggere, scrivere ed eseguire è concesso a tutti quando si crea una nuova directory. Quando invece si crea un nuovo file, tale funzione concederà i permessi di lettura e scrittura a tutti ma non darà i permessi di esecuzione a nessuno per tutte le categorie di utenti. Perciò, prima dell'applicazione della maschera, una directory ha i permessi *777* o *rwxrwxrwx* e un semplice testo *666* o *rw-rw-rw-*.

Il valore di *umask* viene sottratto da questi permessi base dopo che la funzione ha creato il nuovo file o directory. Di conseguenza, se il valore della maschera è *(0)002*, una directory avrà di base i permessi *775* e un file *664*. Ciò viene mostrato nell'esempio seguente.

```
bert:~> mkdir newdir

bert:~> ls -ld newdir
drwxrwxr-x    2 bert     bert         4096 Feb 28 13:45 newdir/

bert:~> touch newfile

bert:~> ls -l newfile
-rw-rw-r--    1 bert     bert            0 Feb 28 13:52 newfile
```

File contro directory
Normalmente una directory ha maggiori permessi: ha sempre il permesso di esecuzione (*execute*). Se non l'avesse, essa sarebbe inaccessibile. Sperimentate ciò eseguendo **chmod *644*** con una directory!

Se vi registrate in un altro gruppo utilizzando il comando **newgrp**, la maschera rimane invariata. Così, se quest'ultima è impostata a *002*, i file e le directory che creerete stando nel nuovo gruppo saranno pure accessibili a tutti gli altri membri di esso: non avrete quindi bisogno di usare **chmod**.

L'utente *root* di base ha dei permessi di creazione dei file più ristretti:

```
[root@estoban root]# umask
022
```

Questi valori di base sono impostati per tutto il sistema nei file di configurazione delle risorse di shell, per esempio `/etc/bashrc` oppure `/etc/profile`. Potete modificarli nel vostro file di configurazione della shell (v. il capitolo 7 dedicato alla personalizzazione del vostro ambiente di shell).

3.4.2.4. Cambiare le proprietà di utente e gruppo

Quando un file è posseduto da un utente o da un gruppo sbagliato, l'errore può essere corretto con i comandi **chown** (change owner) e **chgrp** (change group). Cambiare la proprietà di un file è un compito frequente di amministrazione del sistema in ambienti dove i file hanno la necessità di essere condivisi all'interno di un gruppo. Entrambi i comandi sono molto flessibili, come potete verificare ricorrendo all'opzione `--help`.

Il comando **chown** può essere impiegato per modificare le proprietà dell'utente e del gruppo, mentre **chgrp** cambia solo la proprietà del gruppo. Naturalmente il sistema verificherà se l'utente che fornisce uno di questi comandi ha sufficienti permessi sui file che intende modificare.

Per modificare l'appartenenza di un file ad un utente, usate questa sintassi:

chown ***`newuser`*** **`file`**

Se usate due punti (:) dopo il nome di utente (v. pagine Info), anche l'appartenenza al gruppo verrà cambiata al gruppo primario dell'utente che fornisce il comando. In un sistema Linux ciascun utente ha il proprio gruppo cosicché questo sistema può essere usato per rendere privati i file.

```
jacky:~> id
uid=1304(jacky) gid=(1304) groups=1304(jacky),2034(pproject)

jacky:~> ls -l my_report
-rw-rw-r--  1 jacky   project       29387 Jan 15 09:34 my_report

jacky:~> chown jacky: my_report

jacky:~> chmod o-r my_report

jacky:~> ls -l my_report
-rw-rw----  1 jacky   jacky         29387 Jan 15 09:34 my_report
```

Se *jacky* volesse condividere questo file, senza dare a nessuno il permesso di scriverlo, potrebbe usare il comando **chgrp**:

```
jacky:~> ls -l report-20020115.xls
-rw-rw---- 1 jacky   jacky   45635 Jan 15 09:35 report-20020115.xls

jacky:~> chgrp project report-20020115.xls

jacky:~> chmod o= report-20020115.xls
```

```
jacky:~> ls -l report-20020115.xls
-rw-rw---- 1 jacky   project 45635 Jan 15 09:35 report-20020115.xls
```

In questo modo gli utenti del gruppo *project* potranno lavorare su questo file. Gli utenti estranei al gruppo non avranno nulla a che fare con questo.

Sia **chown** che **chgrp** possono essere usati per cambiare le proprietà ricorsivamente usando l'opzione -R. In tal caso tutti file sottostanti e le sottodirectory di una data directory apparterranno a quel dato utente e/o gruppo.

Restrizioni

In molti sistemi l'utilizzo dei comandi **chown** e **chgrp** è vietato agli utenti non privilegiati. Se non siete gli amministratori di sistemi, non avrete la possibilità di modificare utente e gruppo per ragioni di sicurezza. Se l'uso di questi comandi non fosse limitato, utenti maliziosi potrebbero assegnare la proprietà, modificandola, dei file ad altri utenti e/o gruppi e modificare il comportamento di quegli ambienti utenti e pure causare danni ad altro file degli utenti.

3.4.2.5. Modi speciali

Per evitare che l'amministratore di sistema sia infastidito dal risolvere problemi di permessi per tutto il tempo, possono essere concessi speciali diritti d'accesso a tutte le directory o a programmi separati. Qui di seguito ci sono tre modi speciali:

- modo *sticky bit*: dopo l'esecuzione di un compito, il comando viene mantenuto nella memoria di sistema. In origine questo era una funzione usata spesso per risparmiare memoria: grandi *job* venivano caricati in memoria una sola volta. Ma oggigiorno la memoria è economica e ci sono migliori tecniche per gestirla cosicché il modo non viene più utilizzato per le sue capacità di ottimizzazione su singoli file. Se applicato comunque ad un'intera directory, lo sticky bit assume un diverso significato. In tal caso un utente può modificare file esistenti in questa directory quando è proprietario del file oppure quando il file ha i permessi corretti. Questa funzione viene utilizzata con directory come `/var/tmp`, che deve essere accessibile da chiunque, ma dove non è appropriato per gli utenti modificare o cancellare i dati altrui. Lo *sticky bit* è indicato con una *t* alla fine dell'elenco dei permessi sul file:
  ```
  mark:~> ls -ld /var/tmp
  drwxrwxrwt   19 root     root    8192 Jan 16 10:37 /var/tmp/
  ```
  Lo sticky bit si imposta utilizzando il comando **chmod** ***`o+t directory`***. L'origine storica della "t" si ritrova nella funzione UNIX *save Text access*.
- SUID (set user ID) e SGID (set group ID): rappresentato dalla lettera *s* nel campo dei permessi dell'utente o del gruppo. Quando un file eseguibile ha tale modalità impostata, esso girerà con i permessi dell'utente e del gruppo invece che con quelli dell'utente che batte il comando, dando così acceso alle risorse di sistema. Più avanti discuteremo i ciò nel Capitolo 4.
- SGID (set group ID) su una directory: in questo caso speciale ogni file creato nella directory avrà come gruppo proprietario lo stesso della directory (mentre il normale comportamento sarebbe che i nuovi file fossero di proprietà dell'utente che li crea).

In questa maniera gli utenti non devono preoccuparsi delle proprietà di file quando condividono directory

```
mimi:~> ls -ld /opt/docs
drwxrws---  4 root    users          4096 Jul 25 2001 docs/

mimi:~> ls -l /opt/docs
-rw-rw----  1 mimi    users        345672 Aug 30 2001-Council.doc
```

Questo è il modo normale di condividere file in UNIX.

I file esistenti non vengono modificati!
I file spostati in una directory SGID, ma creati altrove, mantengono i loro utenti e gruppi proprietari. Ciò può confondere.

3.5. Sommario

Con UNIX, come con Linux, tutte le cose sono rappresentate in un modo o in un altro come file con le loro corrette proprietà. L'uso di percorsi (predefiniti) consente agli utenti ed all'amministratore di sistema di trovare, leggere e modificare i file.

Abbiamo compiuto i primi passi per diventare degli esperti: abbiamo trattato della struttura reale e fittizia del file system ed ora sappiamo del modello Linux di sicurezza dei file come pure di diverse altre precauzioni per la sicurezza che vanno adottate normalmente con ogni sistema.

La shell è il più importante strumento per interagire con il sistema. In questo capitolo abbiamo appreso diversi comandi di shell, riportati nella tabella seguente.

Tabella 3-10. Nuovi comandi nel capitolo 3: i file ed il file system

Comando	Significato
bash	Programma GNU di shell.
cat `file(s)`	Invia il contenuto dei file allo standard output
cd `directory`	Entra nella `directory`. **cd** è un comando integrato in **bash**.
chgrp `newgroup file(s)`	Cambia il gruppo proprietario di `file` a *newgroup*
chown mode file	Modifica i permessi di accesso nel `file` .
chmod *newowner*[:[newgroup]] file(s)	Modifica le proprietà di utente e gruppo del file
cp sourcefile targetfile	copia `sourcefile` con il nome `targetfile`
df `file`	Rapporto sullo spazio usato del disco nella partizione contenente `file`.
echo *stringa*	Mostra una linea di testo
export	Parte di **bash** che mostra le variabili ed i loro valori al sistema.
file `nomefile`	Determina il tipo di file di `nomefile`.

Comando	Significato
find ***percorso espressione***	Trova file nella gerarchia del file system
grep ***PATTERN*** **file**	Mostra le linee in `file` che contengono il modello (pattern) di ricerca
head file	Invia la prima parte di `file` allo standard output
id	Stampa il nome e i gruppi dell'utente reale ed attuale
info ***comando***	Mostra la documentazione relativa a **comando**
less file	Mostra `file` con un potente visualizzatore
ln targetfile linkname	Crea un collegamento di nome `linkname` a `targetfile`.
locate ***stringadiricerca***	Mostra tutti i file accessibili che corrispondono al modello di ricerca.
ls file	Stampa il contenuto della directory.
man ***comando***	Formatta e mostra le pagine del manuale (di sistema) in linea relative a **comando**.
mkdir nuovadir	Crea una nuova directory vuota.
mv vecchiofile nuovofile	Rinomina o sposta `vecchiofile`.
newgroup nomegruppo	Registra in un nuovo gruppo
pwd	Mostra la directory di lavoro attuale o corrente.
quota	Mostra l'utilizzo del disco e i limiti.
rm file	Rimuove file e directory.
rmdir file	Rimuove directory.
tail file	Stampa l'ultima parte di `file`.
umask[*valore*]	Mostra o cambia la modalità di creazione dei nuovi file.
wc file	Conta linee, parole e caratteri di `file`.
which ***comando***	Mostra l'intero percorso del **comando**.

Insistiamo pure sul fatto che voi dovreste LEGGERE LE PAGINE MAN. Tale documentazione è il vostro kit di pronto soccorso e contiene le risposte a molte domande. La lista precedente contiene i comandi fondamentali che userete quotidianamente, ma essi possono fare molto di più rispetto a quanto abbiamo detto qui. Leggere la documentazione vi darà il controllo di cui avete bisogno.

Ultima ma non meno importante, ecco un'agile panoramica dei permessi sui file:

Tabella 3-11. Permessi sui file

Chi	r(ead) - lettura	w(rite)-scrittura	(e)x(ecute)
u(ser) - utente	4	2	1
g(roup) - gruppo	4	2	1

Chi	r(ead) - lettura	w(rite)-scrittura	(e)x(ecute)
o(ther) - altri	4	2	1

3.6. Esercizi

Registratevi semplicemente con il vostro ID di utente comune.

3.6.1. Partizioni

- In quale partizione si trova la vostra directory personale?
- Quante partizioni esistono nel vostro sistema?
- Qual è la dimensione totale della vostra installazione di Linux?

3.6.2. Percorsi

- Mostrate il vostro percorso (path) di ricerca.
- Esportate un percorso senza senso inserendo, per esempio, **export** **`PATH=`*`blah`*** e provate ad elencare il contenuto della directory.
- Qual è il percorso della vostra directory personale? Come potrebbe un altro utente raggiungere la vostro directory personale partendo dalla sua e usando un percorso relativo?
- Andate nella directory `tmp` contenuta in `/var`.
- Ora andate in `share` contenuta in `/usr` usando un solo comando. Spostatevi in `doc`. Qual è la vostra attuale directory di lavoro?

3.6.3. Viaggio nel sistema

- Spostatevi nella directory `/proc`.
- Su che CPU sta girando il sistema?
- Quanta RAM sta utilizzando attualmente?
- Quanto spazio di swap avete?
- Quanti driver sono caricati?
- Da quante ore è acceso il sistema?
- Quali filesystem sono riconosciuti dal vostro sistema?
- Spostatevi in `/etc/rc.d | /etc/init.d | /etc/runlevels` e scegliete la directory corrispondente al vostro run level.
- Quali servizi dovrebbero girare in questo livello?
- Quali servizi girano in modo grafico e non in modo testo?
- Spostatevi in `/etc`
- Per quanto tempo il sistema mantiene il file di log in cui sono monitorati i login degli utenti?

- Che versione state facendo girare?
- Ci sono dei messaggi del giorno?
- Quanti utenti sono stati definiti nel vostro sistema? Non contateli, fateglielo fare al computer al vostro posto!
- Quanto gruppi?
- Dove sono tenute le informazioni circa i fusi orari?
- Gli HOWTO sono installati nel vostro sistema?
- Spostatevi in `/usr/share/doc`.
- Citate tre programmi contenuti nel pacchetto GNU *coreutils*.
- Quale versione di **bash** è installata in questo sistema?

3.6.4. Manipolare i file

- Create una nuova directory nella vostra directory personale.
- Potete spostare questa nuova directory allo stesso livello di quella personale?
- Copiate tutti i file XPM da `/usr/share/pixmaps` alla nuova directory. Cosa significa XPM?
- Elencate i file in ordine alfabetico contrario.
- Portatevi nella vostra directory personale. create una nuova directory e copiatevi dentro tutti i file della directory `/etc`. Assicuratevi di copiare pure i file e le sottodirectory di `/etc` (copia ricorsiva)!
- Spostatevi nella nuova directory e create una directory per file che iniziano con un carattere maiuscolo e una per quelli che iniziano con un carattere minuscolo. Spostate tutti i file nelle directory appropriate. Usate meno comandi possibili.
- Rimuovete i file rimanenti.
- Cancellate la directory e il suo intero contenuto utilizzando un unico comando.
- Usate **grep** per trovare quale script avvia il server dei font nel run level grafico.
- Dove si trova il programma server *sendmail*?
- Create un link simbolico nella vostra directory home che punti a `/var/tmp`. Controllate se funziona veramente.
- Create un ulteriore link simbolico nella vostra directory personale al link precedente. Controllate se funziona. Rimuovete il primo link e listate il contenuto della directory. Cosa capita al secondo link?

3.6.5. Permessi dei file

- Riuscite a cambiare i permessi dei file in `/home`?
- Quale è la vostra modalità standard di creazione dei file?
- Cambiate la proprietà di `/etc` con i vostri utente e gruppo personali.
- Cambiate i permessi dei file di `~/.bashrc` in modo che solo voi e il vostro gruppo primario possiate leggerlo.
- Date il comando **locate `root`**. Notate qualcosa di speciale?
- Create un link simbolico a `/root`. Può essere utilizzato?

Capitolo 4. I processi

Dopo i file, i processi sono le cose più importanti in un sistema UNIX/Linux. In questo capitolo daremo uno sguardo da vicino a quei processi. Impareremo di più su:

- Elaborazione multiutente e multitasking
- Tipi di processi
- Controllo dei processi con diversi segnali
- Attributi dei processi
- Il ciclo vitale di un processo
- Avvio e chiusura del sistema
- SUID e SGID
- Velocità e tempi di risposta del sistema
- Programmazione dei processi
- Il sistema Vixie cron
- Come ottenere il massimo dal vostro sistema.

4.1. I processi in dettaglio

4.1.1. Multiutenza e multitasking

Ora che abbiamo acquisito maggiore dimestichezza con il nostro ambiente e siamo un po' capaci di comunicare con il nostro sistema, è giunto il momento di studiare più dettagliatamente i processi (*job*) che possiamo avviare. Non tutti i comandi avviano un unico processo. Alcuni programmi danno inizio ad una serie di processi, come **mozilla**, mentre altri, come **ls**, sono eseguiti come un singolo comando.

D'altra parte Linux si basa su UNIX, dove è pratica comune avere molteplici utenti che fanno girare molti comandi contemporaneamente e nello stesso sistema. E' intuitivo quali misure debbano essere assunte per far sì che la CPU gestisca tutti questi processi e quale funzionalità debba essere fornita in modo che gli utenti possano passare da un processo all'altro. In alcuni casi i processi devono continuare a funzionare anche quando si disconnette l'utente che li ha avviati. Inoltre gli utenti hanno bisogno di un mezzo per riattivare i processi interrotti.

Spiegheremo la struttura dei processi di Linux nelle prossime sezioni.

4.1.2. Tipi di processi

4.1.2.1. Processi interattivi

I processi interattivi si avviano e si controllano per mezzo di una sessione di terminale. In altri termini, deve esserci qualcuno connesso al sistema per poter avviare questi processi: essi non

partono automaticamente come parte delle funzioni di sistema. Tali processi possono girare in primo piano *(foreground*), occupando il terminale che ha avviato il programma, e voi non potete avviare altre applicazioni fintanto che il processo sta girando in primo piano. Altrimenti essi possono girare dietro le quinte (*background*) cosicché il terminale in cui avete avviato il programma può accettare nuovi comandi mentre il programma sta funzionando. Fino ad ora noi ci siamo focalizzati su programmi che girano in primo piano - la durata del tempo richiesto per farli funzionare era talmente breve da essere irrilevante - ma vedere un file con il comando **less** è un buon esempio di comando che impegna la sessione di terminale. In questo caso, il programma avviato sta attendendo da voi di fare qualcosa. Il programma è ancora connesso al terminale da cui era stato avviato ed il terminale è utilizzabile solo per dare comandi che questo programma può comprendere. Altri programmi segnaleranno solo errori o nessun risposta dal sistema.

Comunque, mentre un processo gira sullo sfondo, l'utente può fare altre cose nel terminale in cui ha avviato il programma che sta funzionando.

La shell offre una funzione chiamata *job control* che consente di gestire facilmente più processi. Questo meccanismo seleziona i processi tra primo piano e sfondo. Utilizzando questo sistema i programmi possono essere anche avviati direttamente sullo sfondo.

Far girare un processo sullo sfondo è utile solo con programmi che non necessitano input (tramite shell) da parte dell'utente. Mettere un job in *background* si fa abitualmente quando ci si attende che la sua esecuzione richieda parecchio tempo. Per liberare il terminale dopo l'invio del comando viene aggiunta una E commerciale (&). Nell'esempio, usando la modalità grafica, noi apriamo una finestra di terminale in più a partire da quella esistente:

```
billy:~> xterm &
[1] 26558

billy:~> jobs
[1]+  Running             xterm &
```

Le caratteristiche complete relative al controllo dei processi sono spiegate in modo particolareggiato nelle pagine Info di Bash, cosicché solo le applicazioni di controllo dei job sono elencate qui di seguito:

tabella 4-1. Controllo dei processi

(parte di un) comando	**Significato**
comando_normale	Avvia questo comando in primo piano
comando &	Avvia questo comando sullo sfondo (libera il terminale)
jobs	mostra i comandi che stanno girando sullo sfondo
Ctrl+Z	Sospende (ferma ma non chiude) un processo attivo in primo piano (sospensione).

(parte di un) comando	Significato
Ctrl+C	Interrompe (termina e chiude) un processo che sta girando in primo piano.
%n	Ad ogni processo che gira dietro le quinte (*background*) viene assegnato un numero. Utilizzando la funzione % ci si può riferire ad un job grazie al suo numero, per esempio **fg %2**.
bg	Riattiva un programma sospeso dietro le quinte.
fg	Riporta in primo piano il job.
kill	Termina un programma (v. anche i Comandi interni alla shell nelle pagine Info di **bash**)

Ulteriori esempi pratici si possono trovare tra gli esercizi.

Molti sistemi UNIX sono probabilmente capaci di avviare **screen**, che risulta utile quando volete un'altra shell per eseguire comandi. Dopo aver avviato **screen**, viene creata una nuova sessione con relativa shell e/o comandi come richiesti, che poi potete togliere di mezzo. Nella nuova sessione potete fare qualsiasi cosa. Tutti i programmi e le operazioni gireranno indipendentemente dalla shell di provenienza. Potete poi togliere questa sessione mentre i programmi avviati su di essa continueranno a girare, anche quando vi disconnetterete dalla shell originaria e ripristinerete a piacimento in qualsiasi momento il vostro *screen*.

Tale programma deriva dall'epoca in cui le console virtuali non erano state ancora inventate e tutto andava fatto usando un unico terminale di testo. Per i fanatici è ancora presente in Linux sebbene le console virtuali esistano da almeno dieci anni.

4.1.2.2. Processi automatici

I processi automatici o *batch* non sono collegati ad un terminale. Piuttosto si tratta di operazioni [*task*] che possono essere accodate in un'area di spooler dove attendono di essere eseguite secondo la regola FIFO (first in, first out). Queste possono essere eseguite usando uno dei due criteri:

- ad una certa data e ora: si fa usando il comando **at**, che tratteremo nella seconda parte di questo capitolo.
- nel momento in cui il carico totale del sistema è abbastanza ridotto da accettare lavori extra: si fa usando il comando **batch**. Di base i task vengono accodati in attesa di essere eseguiti quando il carico del sistema è inferiore a 0,8. In grandi ambienti l'amministratore di sistema può preferire l'elaborazione batch quando grandi quantità di dati devono essere elaborate o quando i task che richiedono molte risorse di sistema devono essere eseguiti in un sistema già sovraccarico. L'elaborazione batch è utilizzata anche per ottimizzare le prestazioni del sistema.

4.1.2.3.Daemon

I *daemon* (demoni) sono processi server che girano in continuazione. Il più delle volte vengono inizializzati all'avvio del sistema e poi rimangono in attesa dietro le quinte finché viene richiesto il loro servizio. Un esempio tipico è il demone di rete x*inetd*, che viene fatto partire in quasi tutte le procedure di avvio. Dopo che il sistema si è avviato, il demone di rete “si siede” e attende fino a che un programma cliente, come un cliente FTP, ha bisogno di connettersi.

4.1.3. Attributi dei processi

Un processo possiede una serie di caratteristiche visualizzabili con il comando **ps**:

- Il *process* ID o PID: un unico numero di identificazione usato per riferimenti al processo.
- Il *parent process* ID o PPID: il numero del processo (PID) che ha avviato questo processo.
- Il numero di *nice*: il grado di “cordialità” (*friendliness*) con gli altri processi (da non confondere con la priorità dei processi che viene calcolata in base al numero di nice e al recente utilizzo di CPU da parte del processo).
- Terminale o TTY: terminale a cui è connesso il processo.
- Nome dell'utente reale e effettivo (RUID e EUID): il proprietario del processo. Il proprietario reale è l'utente che ha dato il comando, L'utente effettivo è quello che accede alle risorse di sistema. RUID e EUID normalmente sono la stessa cosa e il processo ha gli stessi diritti di accesso dell'utente che l'ha avviato. Un esempio per chiarire ciò: il browser **mozilla** in `/usr/bin` è di proprietà dell'utente *root*:

```
theo:~> ls -l /usr/bin/mozilla
-rwxr-xr-x  1 root   root       4996 Nov 20 18:28 /usr/bin/mozilla*

theo:~> mozilla &
[1] 26595

theo:~> ps -af
UID      PID  PPID C STIME TTY      TIME CMD
theo   26601  26599 0 15:04 pts/5 00:00:00 /usr/lib/mozilla/mozilla-bin
theo   26613  26569 0 15:04 pts/5 00:00:00 ps -af
```

  Quando l'utente *theo* lancia questo programma, il processo stesso e tutti i processi avviati da questo, saranno di proprietà dell'utente *theo* e non dell'amministratore di sistema. Quando **mozilla** ha necessità di accedere a certi file, quell'accesso sarà determinato dai permessi di *theo* e non da quelli di *root*.
- Gruppo proprietario reale ed effettivo (RGID e EGID): il reale gruppo proprietario di un processo è il gruppo primario dell'utente che ha avviato il processo. L'effettivo gruppo proprietario normalmente è il medesimo, eccetto quando il modo di accesso SGID è stato applicato ad un file.

4.1.4. Visualizzazione delle informazioni sui processi

Il comando **ps** è uno degli strumenti per mostrare i processi. Tale comando ha diverse opzioni che possono essere combinate per mostrare attributi differenti del processo.

Senza opzioni, **ps** dà solo informazioni circa la shell corrente e gli eventuali processi:

```
theo:~> ps
  PID TTY          TIME CMD
 4245 pts/7    00:00:00 bash
 5314 pts/7    00:00:00 ps
```

Dal momento che ciò non vi offre informazioni sufficienti - in genere almeno un centinaio di processi stanno girando nel vostro sistema - di solito selezioneremo processi specifici dalla lista di tutti i processi, usando il comando **grep** in una *pipe* (v. Sezione 5.1.2.1.) come in questa linea con cui sceglieremo e mostreremo tutti i processi posseduti da un particolare utente:

ps -ef | grep *nomeutente*

Questo esempio mostra tutti i processi con nome **bash**, la più comune shell di login nei sistemi Linux:

```
theo:~> ps auxw | grep bash
brenda    31970  0.0  0.3  6080 1556 tty2   S  Feb23   0:00 -bash
root      32043  0.0  0.3  6112 1600 tty4   S  Feb23   0:00 -bash
theo      32581  0.0  0.3  6384 1864 pts/1  S  Feb23   0:00 bash
theo      32616  0.0  0.3  6396 1896 pts/2  S  Feb23   0:00 bash
theo      32629  0.0  0.3  6380 1856 pts/3  S  Feb23   0:00 bash
theo       2214  0.0  0.3  6412 1944 pts/5  S  16:18   0:02 bash
theo       4245  0.0  0.3  6392 1888 pts/7  S  17:26   0:00 bash
theo       5427  0.0  0.1  3720  548 pts/7  S  19:22   0:00 grep bash
```

In questi casi il comando **grep,** che trova linee contenenti la stringa *bash*, viene pure mostrato meglio in sistemi che hanno molto tempo libero. Se non volete che ciò accada, usate il comando **pgrep**.

Le shell Bash sono un caso speciale: questo elenco dei processi mostra quali sono le shell di login (in cui dovete dare nome utente e password, come quando vi connettete in modalità testo o stabilite una connessione remota, al contrario delle shell di non-login, avviate per esempio cliccando su un icona della finestra di terminale). Le shell con login sono precedute da un segno meno (-).

I ?
Spiegheremo l'operatore I nel prossimo capitolo (v. Capitolo 5).

Si possono trovare maggiori informazioni nel solito modo: **ps --help** o **man *ps***. GNU **ps** supporta diversi stili di formato delle opzioni: gli esempi qui sopra non contengono errori.

Notate che **ps** restituisce solo la situazione momentanea dei processi attivi; è una registrazione istantanea. Il programma **top** mostra un immagine più precisa mantenendo aggiornati i risultati forniti da **ps** (con una quantità di opzioni) una volta ogni cinque secondi, generando una nuova lista dei processi che causa periodicamente il più grande carico, mentre integra più informazione sullo spazio di swap in uso e lo stato della CPU, dal file system `proc`:

```
 12:40pm up 9 days, 6:00, 4 users, load average: 0.21, 0.11, 0.03
89 processes: 86 sleeping, 3 running, 0 zombie, 0 stopped
CPU states:  2.5% user,  1.7% system,  0.0% nice, 95.6% idle
Mem:   255120K av, 239412K used, 15708K free, 765K shrd, 22620K buff
Swap: 1050176K av, 76428K used, 973748K free, 82756K cached

  PID USER   PRI NI SIZE  RSS SHARE STAT %CPU %MEM TIME COMMAND
 5005 root   14  0 91572  15M 11580 R    1.9  6.0  7:53 X
```

```
19599 jeff  14  0  1024 1024   796 R    1.1  0.4  0:01 top
19100 jeff   9  0  5288 4948  3888 R    0.5  1.9  0:24 gnome-terminal
19328 jeff   9  0 37884  36M 14724 S    0.5 14.8  1:30 mozilla-bin
    1 root   8  0   516  472   464 S    0.0  0.1  0:06 init
    2 root   9  0     0    0     0 SW   0.0  0.0  0:02 keventd
    3 root   9  0     0    0     0 SW   0.0  0.0  0:00 kapm-idled
    4 root  19 19     0    0     0 SWN  0.0  0.0  0:00 ksoftirqd_CPU0
    5 root   9  0     0    0     0 SW   0.0  0.0  0:33 kswapd
    6 root   9  0     0    0     0 SW   0.0  0.0  0:00 kreclaimd
    7 root   9  0     0    0     0 SW   0.0  0.0  0:00 bdflush
    8 root   9  0     0    0     0 SW   0.0  0.0  0:05 kupdated
    9 root  -1-20     0    0     0 SW<  0.0  0.0  0:00 mdrecoveryd
   13 root   9  0     0    0     0 SW   0.0  0.0  0:01 kjournald
   89 root   9  0     0    0     0 SW   0.0  0.0  0:00 khubd
  219 root   9  0     0    0     0 SW   0.0  0.0  0:00 kjournald
  220 root   9  0     0    0     0 SW   0.0  0.0  0:00 kjournald
```

La prima linea di **top** contiene le stesse informazioni mostrate dal comando **uptime**:

```
jeff:~> uptime
  3:30pm, up 12 days, 23:29, 6 users, load average: 0.01, 0.02, 0.00
```

I dati per questi programmi sono conservati in `/var/run/utmp` (informazione sugli utenti al momento connessi) e nel file system virtuale `/proc`, per esempio `/proc/loadavg` (informazione sul carico medio). Esistono tutti i generi di applicazioni grafiche per vedere questi dati, come lo *Gnome System Monitor* e *lavaps*. Su FreshMeat e SourceForge troverete decine di applicazioni che raccolgono queste informazioni da altri dati dei server e dalle registrazioni provenienti da molteplici server in un unico (web) server, consentendo di monitorare l'intera infrastruttura da una sola workstation.

Le relazioni tra i processi possono essere visualizzate usando il comando **pstree**:

```
sophie:~> pstree
init-+-amd
     |-apmd
     |-2*[artsd]
     |-atd
     |-crond
     |-deskguide_apple
     |-eth0
     |-gdm---gdm-+-X
     |           `-gnome-session-+-Gnome
     |                           |-ssh-agent
     |                           `-true
     |-geyes_applet
     |-gkb_applet
     |-gnome-name-serv
     |-gnome-smproxy
     |-gnome-terminal-+-bash---vim
     |                |-bash
     |                |-bash---pstree
     |                |-bash---ssh
     |                |-bash---mozilla-bin---mozilla-bin---3*[mozilla-bin]
     |                `-gnome-pty-helper
     |-gpm
     |-gweather
     |-kapm-idled
     |-3*[kdeinit]
     |-keventd
     |-khubd
     |-5*[kjournald]
     |-klogd
     |-lockd---rpciod
```

```
     |-lpd
     |-mdrecoveryd
     |-6*[mingetty]
     |-8*[nfsd]
     |-nscd---nscd---5*[nscd]
     |-ntpd
     |-3*[oafd]
     |-panel
     |-portmap
     |-rhnsd
     |-rpc.mountd
     |-rpc.rquotad
     |-rpc.statd
     |-sawfish
     |-screenshooter_a
     |-sendmail
     |-sshd---sshd---bash---su---bash
     |-syslogd
     |-tasklist_applet
     |-vmnet-bridge
     |-xfs
     `-xinetd-ipv6
```

Le opzioni `-u` e `-a` danno delle informazioni ulteriori. Per le altre opzioni e per sapere cosa fanno, ricorrete alle pagine Info.

Nella prossima sezione vedremo come un processo può crearne un altro.

4.1.5. Vita e morte di un processo

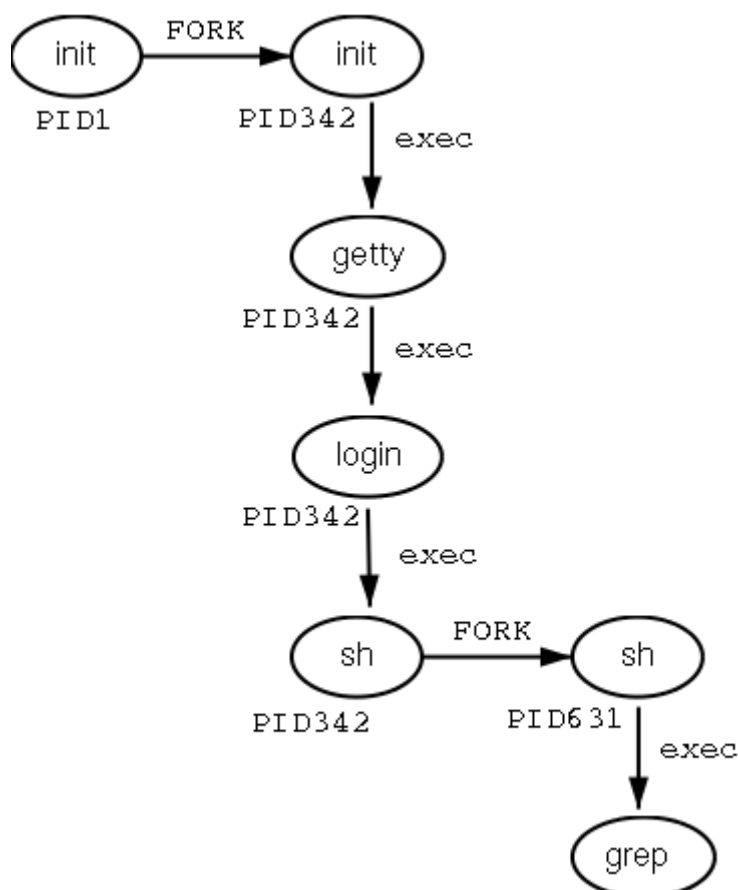
4.1.5.1. Creazione del processo

Un nuovo processo viene creato perché un processo già esistente crea un copia esatta di se stesso. Questo processo figlio (*child*) ha lo stesso ambiente del suo genitore e differisce solo per il diverso numero di process ID (PID). Tale procedura è detta *forking* (biforcazione).

Dopo il processo di biforcazione, lo spazio di indirizzo del processo figlio viene sovrascritto con i dati del nuovo processo: ciò si fa con una chiamata *exec* al sistema.

Poi il meccanismo *fork-and-exec* scambia un vecchio comando con uno nuovo mentre l'ambiente in cui il nuovo programma viene eseguito rimane lo stesso, comprese la configurazione delle periferiche di ingresso e uscita, le variabili d'ambiente e le priorità. Tale meccanismo è impiegato per creare tutti i processi di UNIX, così come è applicato pure nel sistema operativo Linux. Anche il primo processo, **init**, con ID di processo pari a 1, viene sdoppiato (*fork*) durante la procedura di avvio nella cosiddetta procedura di *bootstrap.*

Questo schema illustra il meccanismo di *fork-and-exec*. L'ID di processo cambia dopo la procedura di sdoppiamento:

Figura 4-1. Meccanismo di *fork-and-exec*

Ci sono due casi in cui **init** diventa il genitore di un processo, mentre il processo non era ancora stato avviato da **init** come abbiamo già visto nell'esempio di **pstree**. Molti programmi, per esempio, *demonizzano* i loro processi figlio, in modo che possono mantenersi attivi anche quando il genitore si ferma o viene fermato. Un gestore di finestre è il tipico esempio: esso avvia un processo **xterm** che genera una shell che accetta comandi. Poi il gestore di finestre rifiuta ogni ulteriore responsabilità e passa il processo figlio a **init**. Utilizzando questo meccanismo, è possibile cambiare gestori di finestre senza interrompere le applicazioni attive.

Di tanto in tanto le cose possono andare male, anche nelle migliori famiglie. In casi eccezionali un processo potrebbe finire mentre quello genitore non si aspetta il suo completamento. Così un processo "insepolto" è detto processo *zombie*.

4.1.5.2. Terminare i processi

Quando un processo termina normalmente (non viene cioè ucciso o interrotto in modo inatteso), il programma restituisce il suo *exit status* a quello genitore. Questo exit status è un numero restituito dal programma che fornisce i risultati dell'esecuzione dello stesso. Il sistema di restituire le informazioni riguardanti l'esecuzione dei job ha la sua origine nel linguaggio di programmazione C con cui è stato scritto UNIX.

Il codice di ritorno può essere in seguito interpretato dal processo genitore o da script. I valori dei codici di ritorno sono specifici del programma. Questa informazione può essere normalmente

trovata nelle pagine man dello specifico programma: per esempio il comando **grep** restituisce `-1` se non ci sono state coincidenze, e con ciò si può scrivere un messaggio sulle linee di “No file found”. Un altro esempio è **true**, comando interno a Bash, che non fa altro se non restituire un exit status uguale a 0 per indicare il successo.

4.1.5.3. Segnali

I processi terminano perché ricevono un segnale. Esistono numerosi segnali che potete inviare ad un processo. Utilizzate il comando **kill** per mandare un segnale ad un processo. Il comando **kill `-l`** mostra un elenco di segnali di cui molti sono per uso interno al sistema operativo oppure per programmatori che stanno scrivendo del codice. Come utenti vi serviranno i seguenti segnali:

Tabella 4-2. Segnali comuni

Nome del segnale	Numero del segnale	Significato
SIGTERM	15	Termina il processo in modo ordinato.
SIGINT	2	Interrompe il processo. Un processo può ignorare questo segnale.
SIGKILL	9	Interrompe il processo. Un processo non può ignorare questo segnale.
SIGHUP	1	Per demoni: rilegge il file di configurazione.

Potete leggere di più sulle azioni di base che vengono intraprese con l'invio di un segnale ad un processo in **man *`7 signal`***.

4.1.6. SUID e SGID

Come promesso nel precedente capitolo, ora noi tratteremo delle modalità speciali SUID e SGID in maniera più dettagliata. Tali modalità esistono per fornire ai normali utenti la capacità di eseguire compiti che normalmente non sarebbero in grado di svolgere a causa del rigido schema di permessi sui file utilizzato nei sistemi basati su UNIX. Nella situazione ideale le modalità speciali vengono utilizzate il più raramente possibile dal momento che esse comportano dei rischi di sicurezza. Gli sviluppatori di Linux hanno in genere tentato di evitarle il più possibile, La versione Linux di **ps**, per esempio, utilizza le informazioni contenute nel file system `/proc`, che è accessibile a chiunque, così da evitare l'esposizione di dati e risorse sensibili del sistema al pubblico in genere. Prima di ciò (e ancora nei sistemi UNIX più vecchi) il programma **ps** aveva bisogno di accedere ai file tipo `/dev/mem` e `/dev/kmem`, con gli svantaggi legati ai permessi e alle proprietà di tali file:

```
rita:~> ls -l /dev/*mem
crw-r-----    1 root     kmem       1,   2 Aug 30 22:30 /dev/kmem
crw-r-----    1 root     kmem       1,   1 Aug 30 22:30 /dev/mem
```

Nelle versioni più vecchie di **ps** era impossibile avviare il programma come utente comune, a meno che non gli fossero applicate le modalità speciali.

Mentre in genere tentiamo di evitare l'applicazione di qualsiasi modalità speciale, qualche volta è necessario utilizzare un SUID. Un esempio è costituito dal meccanismo per cambiare le password. Naturalmente gli utenti vorranno fare ciò da sé al posto di avere le loro password impostate dall'amministratore di sistema. Come sappiamo, i nomi utente e le password sono elencate nel file `/etc/passwd`, che ha questi permessi di accesso e proprietari:

```
bea:~> ls -l /etc/passwd
-rw-r--r--    1 root     root      1267 Jan 16 14:43 /etc/passwd
```

Tuttavia gli utenti devono essere capaci di cambiare le propri informazioni contenute in questo file. Ciò si ottiene attribuendo al programma **passwd** dei permessi speciali:

```
mia:~> which passwd
passwd is /usr/bin/passwd

mia:~> ls -l /usr/bin/passwd
-r-s--x--x    1 root     root     13476 Aug  7 06:03 /usr/bin/passwd*
```

Quando viene invocato, il comando **passwd** si avvierà utilizzando i permessi di *root*, abilitando così un comune utente a modificare il file di password che è posseduto dall'amministratore di sistema.

La modalità SGID su di un file non serve così frequentemente come quella SUID, perché spesso SGID implica la creazione di gruppi extra. In alcuni casi, comunque, dobbiamo passare attraverso tale difficoltà per realizzare una soluzione elegante (non preoccupatevi eccessivamente di ciò - i gruppi necessari vengono di solito creati durante l'installazione). Questo è il caso dei programmi **write** e **wall**, che vengono utilizzati per mandare messaggi agli altri terminali d'utenti (tty). Il comando **write** scrive un messaggio ad un singolo utente, mentre **wall** scrive a tutti gli utenti connessi.

Mandare un testo ad un altro terminale d'utente o ad uno schermo grafico non è normalmente permesso. Per superare questo problema è stato creato un gruppo che possiede tutte le periferiche di terminale. Quando i comandi **write** e **wall** hanno i permessi SGID, essi gireranno con i diritti di accesso di questo gruppo, *tty* nell'esempio. Dal momento che tale gruppo ha l'accesso in scrittura nel terminale di destinazione, anche un utente che non ha il permesso di usare quel terminale in qualche maniera gli può inviare un messaggio.

Nell'esempio seguente, l'utente *joe* prima cerca con il comando **who** in quale terminale la sua corrispondente è connessa, poi le invia un messaggio usando il comando **write**. Sono pure illustrati i diritti d'accesso per il programma **write** e sui terminali occupati dalla utente ricevente: è evidente che altri, all'infuori dell'utente proprietario, non hanno i permessi su quella periferica ad eccezione del gruppo proprietario, che può scrivere su di essa.

```
joe:~> which write
write is /usr/bin/write

joe:~> ls -l /usr/bin/write
-rwxr-sr-x    1 root     tty       8744 Dec  5 00:55 /usr/bin/write*

joe:~> who
jenny     tty1     Jan 23 11:41
```

```
jenny     pts/1    Jan 23 12:21 (:0)
jenny     pts/2    Jan 23 12:22 (:0)
jenny     pts/3    Jan 23 12:22 (:0)
joe       pts/0    Jan 20 10:13 (lo.callhost.org)

joe:~> ls -l /dev/tty1
crw--w----    1 jenny   tty  4,     1 Jan 23 11:41 /dev/tty1

joe:~> write jenny tty1
hey Jenny, pranziamo assieme?
^C
```

L'utente *jenny* riceve questo sul suo schermo:

```
Message from joe@lo.callhost.org on ptys/1 at 12:36 ...
hey Jenny, pranziamo assieme?
EOF
```

Dopo aver ricevuto un messaggio, il terminale può essere cancellato usando la combinazione di tasti **Ctrl+L**. Per non ricevere alcun messaggio (eccetto che dall'amministratore di sistema) usate il comando **mesg**. Per vedere quali utenti connessi accettano messaggi da altri usate **who -w**. Tutte le caratteristiche sono spiegate nelle pagine Info di ciascun comando.

I nomi dei gruppi possono variare
Lo schema dei gruppi è specifico di ogni distribuzione. Altre distribuzioni possono utilizzare altri nomi o soluzioni.

4.2. Processo d'avvio, init e shutdown

4.2.1. Introduzione

Uno dei più potenti aspetti di Linux riguarda il suo metodo aperto di avviare e chiudere il sistema operativo, dove carica specifici programmi utilizzando le loro particolari configurazioni, vi permette di cambiare tali configurazioni per controllare il processo d'avvio e chiude in un modo pulito ed organizzato.

Oltre alla questione del controllo del processo d'avvio o di chiusura, la natura aperta di Linux rende più semplice determinare la fonte esatta di molti problemi associati agli stessi. Una infarinatura di questi processi è abbastanza utile per chiunque utilizzi un sistema Linux.

Molte macchine Linux adoperano **lilo**, il LInux LOader, per avviare il sistema operativo. Noi tratteremo comunque solo di GRUB che è più facile da usare e maggiormente flessibile. Se avete bisogno di informazioni su **lilo**, potete riferirvi alle pagine man e agli HOWTO. Entrambi i sistemi supportano installazioni in dual boot (faremo riferimento agli HOWTO su questo argomento per esempi pratici e informazioni di fondo).

4.2.2. Il processo di avvio

Quando si avvia un computer x86, il processore ricerca il BIOS (Basic Input/Output System) alla fine della memoria di sistema e lo esegue. Normalmente quest'ultimo controlla il lettore dei floppy (o il CD-ROM in molti dei sistemi più recenti) alla ricerca di un supporto avviabile, se presente, e successivamente cerca nel disco rigido. L'ordine delle periferiche usate per il cosiddetto *boot* di solito è controllato da una specifica impostazione del BIOS di sistema. Una volta installato Linux nel disco rigido di un computer, il BIOS cerca un *Master Boot Record* (MBR) posizionato nel primo settore del primo disco fisso, carica il suo contenuto in memoria e poi gli passa il controllo.

Questo MBR contiene istruzioni su come caricare il boot-loader GRUB (o LILO), che usa un sistema operativo preselezionato. Quindi MBR carica il boot-loader, il quale assume a sua volta il controllo del processo (se, naturalmente, il boot-loader è installato in MBR). Nella configurazione base di Red Hat Linux GRUB usa le impostazioni contenute in MBR per mostrare in un menu le opzioni di avvio. Una volta che GRUB ha ricevuto le corrette istruzioni per il sistema operativo da far partire (sia dalla sua linea di comando che dal file di configurazione), esso trova il file di boot necessario e lascia il controllo della macchina a quel sistema operativo.

4.2.3. Caratteristiche di GRUB

Questo metodo di avvio è detto caricamento diretto (*direct loading*) perché sono usate delle istruzioni per avviare direttamente il sistema operativo, senza codice intermedio tra i boot-loader e i principali file del sistema operativo (come il kernel). Il processo di boot impiegato da altri sistemi operativi può comunque differire leggermente da quello sopra descritto. Per esempio, i sistemi operativi Microsoft DOS e Windows sovrascrivono qualsiasi cosa in MBR quando vengono installati senza incorporare alcunché della precedente configurazione di MBR: ciò distrugge ogni altra informazione scritta in MBR da altri sistemi operativi, come ad esempio Linux. I sistemi operativi Microsoft, come anche vari altri sistemi operativi proprietari, vengono caricati con un metodo di avvio a caricamento in catena (*chain loading boot method*). Con questo metodo MBR punta al primo settore della partizione contenente il sistema operativo, dove trova i file speciali necessari per avviare effettivamente tale sistema operativo.

GRUB supporta entrambi i metodi di boot, consentendovi di utilizzarlo con quasi tutti i sistemi operativi, con molti file system e qualsiasi disco fisso riconosciuto dal vostro BIOS.

GRUB possiede numerose altre caratteristiche tra cui (le più importanti):

- GRUB fornisce un vero ambiente pre-Sistema Operativo, basato su comandi, per macchine x86 che consente la massima flessibilità nel caricare sistemi operativi con certe opzioni o nel raccogliere informazioni sul sistema.
- GRUB supporta la modalità di indirizzamento logico dei blocchi (LBA o *Logical block Addressing*), necessaria per accedere a molti dischi fissi IDE e a tutti quelli SCSI. Prima di LBA i dischi fissi potevano andare incontro al limite del cilindro 1024 oltre al quale il BIOS poteva non trovare un file.
- Il file di configurazione di GRUB viene letto dal disco ogni volta che il sistema si

avvia, evitandovi di dover scrivere nel MBR tutte le volte che cambiate le opzioni di boot.

Una descrizione completa di GRUB si può avere con il comando **info *grub*** oppure nel sito di GRUB. Il Progetto di Documentazione Linux ha il Multiboot with GRUB Mini-HOWTO.

4.2.4. Init

Il kernel, dopo essere stato caricato, trova **init** in `sbin` e lo esegue.

Quando parte **init**, esso diviene il genitore o il nonno di tutti i processi che si avviano automaticamente nel vostro sistema Linux. La prima cosa che **init** fa è leggere il suo file di inizializzazione, `/etc/inittab`. Quest'ultimo istruisce **init** a leggere uno script iniziale di configurazione dell'ambiente che imposta il path, avvia lo swapping, controlla i file system, e così via. Fondamentalmente questa fase si cura di tutte le cose di cui ha bisogno il vostro sistema al momento della inizializzazione: impostare l'orologio, inizializzare le porte seriali, ecc.

Poi **init** continua la lettura del file `/etc/inittab` che descrive come il sistema dovrebbe essere impostato in ciascun run level (livello di esecuzione o avvio) e seleziona quello di partenza. Un run level è una configurazione di processi. Tutti i sistemi simil-UNIX possono essere avviati con diverse configurazioni di processi, come la modalità utente singolo, che viene definita come run level 1 o S (o s). In questa modalità solo l'amministratore di sistema può connettersi al sistema: viene utilizzata per attività di manutenzione senza rischi di danni al sistema o ai dati degli utenti. Naturalmente con questa configurazione non abbiamo bisogno di offrire servizi di utente, cosicché essi saranno tutti disabilitati. Un altro run level è quello di reboot (o run level 6) che termina tutto i servizi attivi seguendo le appropriate procedure e poi riavvia il sistema.

Usate **who** per controllare qual'è il vostro attuale livello d'esecuzione:

```
willy@ubuntu:~$ who -r
         run-level 2 2006-10-17 23:22 last=S
```

Di più sui livelli di esecuzione [run level] nella prossima sezione (v. Sezione 4.2.5).

Dopo aver stabilito l'iniziale livello di esecuzione per il vostro sistema, **init** lancia tutti i processi di background necessari per far girare il sistema cercando nella directory `rc` specifica di quel run level. **init** avvia ogni script killer (i loro nomi di file iniziano con K) con un parametro di stop. Dopo fa girare tutti gli script di partenza (i loro nomi di file iniziano con S, cioè Start) contenuti nella directory del corrispondente livello di avvio in modo che tutti i servizi e le applicazioni vengano lanciate correttamente. Di fatto, dopo che il sistema ha terminato l'avvio, potete eseguire manualmente questi stessi script con un comando tipo **/etc/init.d/httpd *stop*** o **service *httpd stop*** connessi come *root* (fermando in questo caso il web server).

Caso speciale
Notate che all'avvio del sistema, gli script in `rc2.d` e `rc3.d` vengono eseguiti

normalmente. In questo caso, nessun servizio viene fermato (almeno non permanentemente). Ci sono solo servizi che vengono attivati.

Nessuno degli script che normalmente avviano e fermano i servizi sono collocati in /etc/rc<x>.d. Piuttosto tutti i file in /etc/rc<x>.d sono collegamenti simbolici che puntano agli script reali posizionati in `/etc/init.d`. Un collegamento simbolico non è altro se non un file che punta ad un altro file ed in tal caso è utilizzato perché può essere creato ed eliminato senza influire sugli script reali che uccidono o avviano i servizi. I collegamenti simbolici a vari script sono numerati in un ordine particolare che determina la sequenza di avvio. Potete cambiare l'ordine con cui si avviano e si uccidono i servizi modificando il nome del collegamento simbolico che si riferisce allo script che realmente controlla lo script. Potete utilizzare lo stesso numero più volte se volete che un particolare servizio sia fermato o avviato prima o dopo di un altro, come nell'esempio seguente che elenca il contenuto di `/etc/rc5.d`, directory in cui **crond** e **xfs** sono entrambi avviati da un nome di link che inizia per "S90". In questo caso gli script vengono avviati in ordine alfabetico.

```
[jean@blub /etc/rc5.d] ls
K15httpd@     K45named@    S08ipchains@  S25netfs@      S85gpm@
K16rarpd@     K46radvd@    S08iptables@  S26apmd@       S90crond@
K20nfs@       K61ldap@     S09isdn@      S28autofs@     S90xfs@
K20rstatd@    K65identd@   S10network@   S30nscd@       S95anacron@
K20rusersd@   K74ntpd@     S12syslog@    S55sshd@       S95atd@
K20rwalld@    K74ypserv@   S13portmap@   S56rawdevices@ S97rhnsd@
K20rwhod@     K74ypxfrd@   S14nfslock@   S56xinetd@     S99local@
K25squid@     K89bcm5820@  S17keytable@  S60lpd@
K34yppasswdd@  S05kudzu@    S20random@    S80sendmail@
```

Dopo che **init** è passato per i livelli di avvio fino a raggiungere quello predefinito, lo script `/etc/inittab` biforca un processo **getty** per ciascuna console virtuale (richiesta di login in modo testo). **getty** apre linee tty, imposta le loro modalità, presenta la richiesta di login, ottiene il nome utente e poi dà inizio al processo di connessione di quell'utente, Tutto ciò consente agli utenti di autenticarsi nel sistema e di usarlo. Per definizione, molti sistemi offrono sei console virtuali, ma, come potete riscontrare nel file `inittab`, ciò è configurabile.

`/etc/inittab` può anche dire a **init** come gestire la pressione da parte dell'utente dei tasti **Ctrl+Alt+Del** nella console. Siccome il sistema dovrebbe essere spento e riavviato con le dovute maniere piuttosto che con l'immediato spegnimento dell'energia elettrica, a **init** viene detto di eseguire il comando **/sbin/shutdown `-t3 -r` *now***, per esempio, quando l'utente preme quei tasti. In aggiunta, `/etc/inittab` stabilisce cosa **init** deve fare in caso di interruzione dell'alimentazione se il vostro sistema è dotato di unità UPS.

In molti sistemi basati su RPM la videata grafica di login viene avviata nel run level 5, quando `/etc/inittab` avvia lo script `/etc/X11/prefdm`. Tale script lancia il gestore dello schermo (display manager) di X preferito, basato sui contenuti della directory `/etc/sysconfig/desktop`. Normalmente si tratta di **gdm** se lanciate GNOME o di **kdm** se invece avviate KDE, ma essi possono essere combinati e c'è pure **xdm** che fa parte dell'installazione standard di X.

Esistono però anche altre possibilità. Per esempio sotto Debian c'è un initscript per ciascuno dei

gestori di schermo e il contenuto di `/etc/X11/default-display-manager` viene usato per stabilire quale di questi avviare. E' possibile leggere qualcosa di più sull'interfaccia grafica nella Sezione 7.3. Per finire, la documentazione di sistema spiegherà dettagliatamente gli aspetti di più alto livello di **init**.

Le directory `/etc/default` e/o `/etc/sysconfig` contengono dati su una serie di funzioni e servizi che vengono letti al momento del boot. La posizione della directory contenente le impostazioni di base potrebbe essere in qualche modo diversa a seconda della vostra distribuzione Linux.

Oltre l'ambiente grafico dell'utente, anche una grande quantità di altri servizi può essere avviata. Se tutto procede per il meglio, dovreste vedere una richiesta di autenticazione o una schermata di login al termine del processo di boot.

Altre procedure

Abbiamo spiegato come **init** SysV opera nei computer basati su x86. Le procedure di avvio possono variare a seconda delle architetture e delle distribuzioni. Altri sistemi usano **init** stile BSD in cui i file di avvio non sono suddivisi in molteplici directory `/etc/rc<LIVELLO>`. E' anche possibile che il vostro sistema usi `/etc/rc.d/init.d` al posto di `/etc/init.d`.

4.2.5. I livelli di esecuzione di init

L'idea base dei differenti servizi operanti nei diversi livelli di esecuzione (o **run level**) è fondata sul concetto che sistemi diversi possono essere utilizzati in modi diversi. Non si usano certi servizi fino a che il sistema non si trovi in uno stato (o modo) particolare come, ad esempio, quando è disponibile per più utenti o per il collegamento in rete.

Esistono occasioni in cui potreste voler usare il sistema in una modalità di livello più basso. Esempi sono la sistemazione dei problemi di corruzione dei dischi a livello 1, in modo che nessun altro utente si trovi nel sistema, oppure il lasciare un server nel livello di esecuzione 3 senza una sessione di X in funzione. In questi casi non ha senso far girare servizi che dipendono da una modalità di sistema più alta in quanto, comunque, non lavorerebbero correttamente. Avendo già assegnato a ciascun servizio di avviarsi quando si è raggiunto il suo specifico livello di esecuzione, voi garantite un ordinato processo di avvio e potete rapidamente cambiare la modalità della macchina senza preoccuparvi di quale servizio avviare o terminare manualmente.

I livelli di esecuzione disponibili si trovano generalmente descritti in `/etc/inittab`, che vi mostriamo parzialmente qui di seguito:

```
#
# inittab   This file describes how the INIT process should set up
#           the system in a certain run-level.

# Default runlevel. The run levels are:
#   0 - halt (Do NOT set initdefault to this)
#   1 - Single user mode
#   2 - Multiuser, without NFS
```

```
#       (The same as 3, if you do not have networking)
#   3 - Full multiuser mode
#   4 - unused
#   5 - X11
#   6 - reboot (Do NOT set initdefault to this)
#
id:5:initdefault:
<--cut-->
```

Sentitevi liberi di configurare i *run level* inutilizzati (comunemente il 4) come meglio vi aggrada. Molti utenti configurano questi livelli di esecuzione in modo da ottimizzarli per sé al massimo, mentre lasciano i livelli di esecuzione 3 e 5 standard. Ciò consente loro di spostarsi rapidamente dentro e fuori dalla configurazione personale senza interferire con il normale insieme di funzioni dei livelli standard.

Se il vostro computer cade in uno stato in cui non può avviarsi a causa di un errato `/etc/inittab` o non vi lascia autenticarvi a causa di un file `/etc/passwd` corrotto (o se avete semplicemente dimenticato la password), fatelo partire nella modalità "utente singolo" (run level 1).

Niente grafica?
Quando lavorate in modalità grafica perché non avete avuto la richiesta di autenticazione grafica sulla console del vostro computer, di solito potete passare alla console 7 (o maggiore) per ottenere un login grafico. Se questo non è il vostro caso, verificate il livello di esecuzione corrente utilizzando il comando **who -r**. Se è impostato su qualcosa di diverso da quello standard originale da `/etc/inittab`, è probabile che il sistema non si avvii nella normale modalità grafica: in tal caso contattate l'amministratore di sistema o leggetevi **man init**. Osservate che il passaggio di livello avviene utilizzando preferibilmente il comando **telinit**: passare da una console testuale ad una grafica o viceversa non comporta un cambio di livello di esecuzione.

In questa guida la trattazione dei livelli di avvio, script e configurazioni cerca di essere il più generica possibile anche se esistono molte differenze. Per esempio, Gentoo Linux conserva gli script in `/etc/run levels`. Altri sistemi potrebbero partire attraverso uno o più livelli minori ed eseguire tutti i relativi script prima di arrivare al livello di esecuzione definitivo e di eseguire quegli script. Fate riferimento alla vostra documentazione di sistema per maggiori informazioni. Potreste pure scorrervi gli script richiamati da `/etc/inittab` per acquisire una migliore comprensione di cosa avviene nel vostro sistema

4.2.5.1. Strumenti

I programmi di utilità **chkconfig** o **update-rc.d**, se installati nel vostro sistema, forniscono un semplice strumento a riga di comando per la manutenzione della gerarchia della directory `/etc/init.d`: essi sollevano gli amministratori di sistema dal dover manipolare direttamente i numerosi collegamenti simbolici delle directory sotto `/etc/rc[x].d`.

Inoltre, alcuni sistemi offrono lo strumento **ntsysv**, che fornisce un'interfaccia testuale (potrete trovare questa più facile da usare dell'interfaccia a riga di comando di **chkconfig**). Con SuSE Linux, avrete gli strumenti **yast** e **insserv**. Per la configurazione semplificata di Mandrake, potreste vole provare *DrakConf*, che consente, fra le altre funzionalità, di passare dal livello di esecuzione 3 al 5.

Con Mandriva questo diventa il Mandriva Linux Control Center.

Molte distribuzioni offrono un'interfaccia utente grafica per configurare i processi: verificate nella vostra documentazione di sistema.

Tutte queste utility devono essere avviate come root. L'amministratore di sistema può anche creare manualmente gli appropriati collegamenti in ogni directory di run level per avviare o fermare un servizio di un certo livello di esecuzione.

4.2.6. Lo spegnimento

UNIX non è stato creato per essere spento, ma se proprio dovete, utilizzate il comando **shutdown**. Dopo il completamento della procedura di spegnimento, l'opzione `-h` fermerà il sistema, mentre `-r` lo riavvierà.

I comandi **reboot** e **halt** ora sono capaci di invocare **shutdown** se lanciati quando il sistema si trova nei livelli di esecuzione da 1 a 5 - e così si assicura un corretto spegnimento (*shutdown*) - ma si tratta di una pessima abitudine da acquisire e non tutte le versioni UNIX/Linux hanno questa funzionalità.

Se il vostro computer non si spegne da solo, non dovreste farlo fino a che non vedete un messaggio che vi indica che il sistema è fermo o che la chiusura non è terminata, per dare tempo al sistema di smontare tutte le partizioni. Essere impazienti potrebbe causare una perdita di dati.

4.3. La gestione dei processi

4.3.1. Lavori per l'amministratore di sistema

Mentre gestire le risorse di sistema, compresi i processi, è un compito dell'amministratore del sistema locale, non nuoce ad un comune utente conoscere qualcosa di questo, specialmente quando ciò riguarda i suoi processi e la loro esecuzione ottimale.

Spiegheremo un po' a livello teorico le prestazioni del sistema, sebbene non fino alla ottimizzazione dell'hardware e simili. Piuttosto studieremo i problemi quotidiani con cui deve confrontarsi un comune utente e le azioni da intraprendere in tale qualità per usare al meglio le risorse disponibili. Come impareremo nella prossima sezione, ciò costituisce un motivo per pensare prima di agire.

Figura 4-2. Puoi correre più velocemente?

4.3.2. Quanto tempo richiede?

Bash offre il comando integrato **time** che mostra quanto tempo un comando impiega per essere eseguito. Il cronometraggio è altamente accurato e può essere usato con qualsiasi comando. Nell'esempio seguente esso impiega un minuto e mezzo per compilare questo testo:

```
tilly:~/xml/src> time make
Output written on abook.pdf (222 pages, 1619861 bytes).
Transcript written on abook.log.

real    1m41.056s
user    1m31.190s
sys     0m1.880s
```

Il comando GNU **time** in `/usr/bin` (a differenza della versione integrata nella shell) mostra maggiori informazioni presentabili in diversi modi. Mostra anche lo stato di uscita del comando ed il tempo totale trascorso. Lo stesso comando che usa il **time** indipendente appena descritto dà questo risultato a video:

```
tilly:~/xml/src> /usr/bin/time make
```

```
Output written on abook.pdf (222 pages, 1595027 bytes).
Transcript written on abook.log.

Command exited with non-zero status 2
88.87user 1.74system 1:36.21elapsed 94%CPU
                                (0avgtext+0avgdata 0maxresident)k
0inputs+0outputs (2192major+30002minor)pagefaults 0swaps
```

Riportatevi nuovamente alle pagine Info per tutte le informazioni.

4.3.4. Le prestazioni

Per un utente "prestazioni" significa una rapida esecuzione dei comandi. Per un gestore di sistema, d'altro canto, significa molto di più: l'amministratore deve ottimizzare le prestazioni del sistema complessivo, compresi gli utenti, tutti i programmi e i demoni. Le prestazioni del sistema possono dipendere da un migliaio di piccole cose che non vengono considerate dal comando **time**:

- il programma in esecuzione è stato scritto male o non usa adeguatamente il computer
- l'accesso ai dischi, ai controller, al display, a tutti i tipi d'interfaccia, ecc...
- la raggiungibilità dei sistemi remoti (prestazioni di rete)
- la quantità di utenti nel sistema e di quelli che stanno lavorando in contemporanea
- il momento della giornata
- ...

4.3.4. Il carico

In breve: il carico dipende da ciò che è normale per il vostro sistema. Il mio vecchio P133, che sta eseguendo un firewall, un server SSH, un file server, un demone di route, un server sendmail, un server proxy e alcuni altri servizi, non si lamenta con 7 utenti connessi: il carico è ancora mediamente 0. Alcuni sistemi (multi-CPU) che ho visto erano invece abbastanza contenti con un carico pari a 67. Esiste solo un modo per scoprirlo: controllatelo regolarmente se volete sapere se è normale. Se non lo fate, sarete solo capaci di misurare il carico di sistema dal tempo di risposta della linea di comando, che è una misurazione molto difficile dal momento che questa velocità è influenzata da un centinaio di altri fattori.

Tenete in mente che differenti sistemi si comporteranno in maniera diversa con la stessa media di carico. Per esempio, un computer con una scheda grafica che supporta l'accelerazione grafica hardware non incontrerà problemi nella creazione di immagini 3D, mentre lo stesso computer con una scheda VGA economica rallenterà spaventosamente. Il mio vecchio P133 diverrà abbastanza scomodo avviando il server X, ma in un moderno computer noterete pesantemente la differenza nel carico di sistema.

4.3.5. Posso fare qualcosa come utente?

Un grosso ambiente può rallentarvi. Se avete molte variabili d'ambiente (al posto di variabili di shell), percorsi di ricerca lunghi e non ottimizzati (errori nell'impostazione della variabile

ambientale del percorso) e così via, il sistema avrà bisogno di più tempo per ricercare e leggere dati.

In X, i gestori delle finestre e gli ambienti desktop possono essere dei veri “mangiatori di CPU”. Un desktop realmente estroso comporta dei costi, anche quando lo scaricate gratis, dal momento che la maggior parte dei desktop si può dotare di programmi aggiuntivi *ad infinitum*. La modestia è una virtù, se non comprate un nuovo computer ogni anno.

4.3.5.1. La priorità

La priorità, ovvero l'importanza di un processo, è definita dal numero di *nice*. Un programma con un numero di *nice* alto è conciliante con gli altri programmi, gli altri utenti e il sistema; non è un job importante. Minore è il numero di nice e più importante è un job ed esso richiederà maggiori risorse senza condividerle.

L'incremento del numero di nice ad un job è utile soltanto nel caso di processi che utilizzano pesantemente il tempo della CPU (compilatori, applicazioni matematiche e così via). I processi che usano sempre molto tempo di I/O vengono automaticamente ricompensati dal sistema con concessione di una priorità maggiore (un numero inferiore di nice): per esempio l'input da tastiera ottiene sempre la più alta priorità in un sistema.

Per definire la priorità di un programma si ricorre al comando **nice**.

Molti sistemi forniscono anche il comando BSD **renice**. che vi consente di variare il valore nice di un comando in esecuzione. Nuovamente, leggete la pagina man per specifiche informazioni sul sistema.

> **Programmi interattivi**
> Non è una buona idea utilizzare **nice** o **renice** su un programma interattivo o su un programma in esecuzione in primo piano.

L'impiego di questi comandi è abitualmente un compito dell'amministratore di sistema. Leggete le pagine man per maggiori informazioni sulle funzionalità extra a disposizione dell'amministratore di sistema.

4.3.5.2. Le risorse della CPU

In ogni sistema Linux molti programmi vogliono usare la (le) CPU contemporaneamente, anche se vi è un unico utente del computer. Ogni programma necessita di una certa quantità di cicli di CPU per poter girare. Potrebbero esistere delle volte in cui non ci sono abbastanza cicli a causa della CPU troppo occupata. Il comando **uptime** è largamente impreciso (fornisce solo delle medie, dovete capire che è normale), ma lontano dall'essere inutile. Esistono alcune operazioni che potete intraprendere se ritenete che la CPU sia da incolpare per l'inerzia del vostro sistema:

- Avviare programmi pesanti quando il carico è basso. Questo potrebbe essere il caso del vostro sistema in tempo di notte (v. la prossima sezione per la programmazione

degli eventi).
- Evitare al sistema lo svolgimento di lavoro non necessario: fermate demoni e programmi non utilizzati, usate **locate** al posto di un pesante **find**, ...
- Avviate grossi job con una priorità bassa

Se nessuna di queste soluzioni è utilizzabile nella vostra situazione specifica, dovrete considerare l'aggiornamento della CPU. In una macchina UNIX questo è un compito dell'amministratore di sistema.

4.3.5.3. Le risorse di memoria

Quando i processi attualmente in funzione si aspettano più memoria di quella fisicamente disponibile nel sistema, un sistema Linux non andrà in *crash*: esso comincerà a impaginare o a *swappare*, cioè ad usare la memoria su disco o nell'area di swap, spostando il contenuto della memoria fisica (pezzi di programmi in funzione o interi programmi in caso di swap) nel disco, in modo da sgombrare la memoria fisica per gestire più processi. Il comando **top** può essere utilizzato per mostrare l'uso della memoria e della swap. I sistemi che usano glibc offrono i comandi **memusage** e **memusagestat** per visualizzare l'uso della memoria.

Se scoprite che molta della memoria e dello spazio sono usati, potete provare a:

- uccidere, fermare o cambiare la priorità di quei programmi che usano una grossa quantità di memoria
- aggiungere più memoria (ed in alcuni casi più spazio di swap) al computer
- aggiustare le prestazioni del sistema, argomento che va al di là dello scopo di questo documento (v. l'elenco di letture in Appendice A per maggiori informazioni).

4.3.5.4. Le risorse di I/O

Mentre le limitazioni di I/O (cioè Input/Output o Ingresso/Uscita) sono una delle principali cause di stress per gli amministratori di sistema, Linux offre utility piuttosto povere per misurare le prestazioni di I/O. Gli strumenti **ps, vmstat** e **top** danno qualche indicazione su quanti programmi siano in attesa di I/O; **netstat** mostra le statistiche dell'interfaccia di rete, ma non esistono realmente strumenti disponibili per misurare la reattività dell'I/O al carico di sistema ed il comando **iostat** restituisce una breve descrizione dell'uso dell'I/O in generale. Esistono vari front-end grafici per rappresentare l'output di tali comandi in una forma umanamente comprensibile.

Ciascuna periferica ha i suoi propri problemi, ma le larghezze di banda a disposizione delle interfacce di rete e dei dischi sono le due cause primarie dei colli di bottiglia nelle prestazioni di I/O.

I problemi di I/O della rete:

- Rete sovraccarica:
  la quantità di dati trasportati sulla rete è maggiore della sua capacità, con conseguente esecuzione lenta di qualsiasi operazione di rete per tutti gli utenti. Ciò si

può risolvere ripulendo la rete (cosa che implica la disabilitazione di protocolli e servizi non necessari) o riconfigurando la rete (per esempio con il ricorso a sottoreti, alla sostituzione degli hub con switch, all'aggiornamento di interfacce e attrezzature);
- Problemi di integrità della rete:
  capitano quando i dati vengono trasferiti in modo scorretto. Per risolvere questo tipo di problema si può solamente isolare l'elemento guasto e sostituirlo.

Problemi di I/O del disco:
- rateo di trasferimento per processo eccessivamente lento:
  non è sufficiente la velocità di lettura o scrittura per un singolo processo;
- rateo aggregato del trasferimento troppo lento;
  la larghezza di banda totale massima che il sistema può fornire a tutti i programmi non è sufficiente.

Questo tipo di problemi è più difficile da individuare e normalmente richiede dell'hardware extra per redistribuire i flussi di dati nei bus, controller e dischi se il sovraccarico dell'hardware ne è l'origine. Una soluzione di ciò è la configurazione di un sistema RAID ottimizzato per le operazioni di input e output. In questo modo manterrete lo stesso hardware. L'altra opzione è l'aggiornamento a bus, controller e dischi più veloci.

Se la causa non è il sovraccarico, potrebbe trattarsi del vostro hardware che si sta progressivamente deteriorando o della sua non perfetta connessione al sistema. Per cominciare, controllate contatti, connettori e prese.

4.3.5.5. Gli utenti

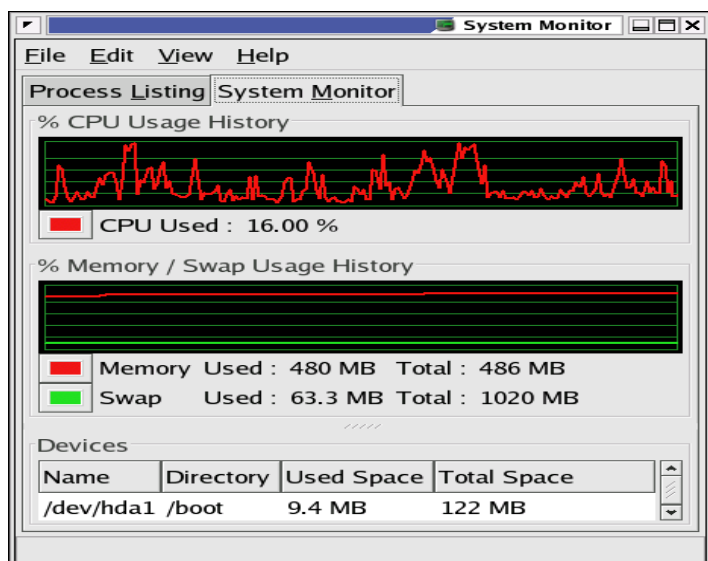
Gli utenti si possono suddividere in diverse categorie in base al loro comportamento nell'utilizzo delle risorse:

- utenti che avviano un (grande) numero di piccoli processi: voi, per esempio, nuovi utenti Linux;
- utenti che avviano relativamente pochi processi ma di grosse dimensioni: simulazioni, calcoli, emulatori o altri programmi che divorano una quantità di memoria e solitamente accompagnati da grandi file d dati;
- utenti che eseguono pochi processi ma usano molto tempo della CPU (sviluppatori e simili).

Potete constatare che le richieste di sistema possono variare a seconda della classe di utenti e che è difficile accontentare tutti. Se vi trovate in un sistema multiutente, è utile (e divertente) scoprire le abitudini degli altri utenti nel sistema per ottenere il massimo per i vostri impieghi specifici.

4.3.5.6. Strumenti grafici

Per gli ambienti grafici sono disponibili strumenti di monitoraggio in quantità. Qui sotto c'è l'immagine del Monitor di Sistema Gnome, che possiede le caratteristiche di mostrare e ricercare informazioni sui processi e di monitorare le risorse di sistema:

Figura 4-3. Il Monitor di Sistema Gnome

C'è anche un paio di comode icone che potete installare nella task bar come monitor di dischi, memoria e carico. **xload** è un'altra piccola applicazione di X per il controllo del sistema. Scoprite la vostra preferita!

4.3.5.7. L'interruzione dei vostri processi

Come utenti non privilegiati, voi potete influire solo sui vostri processi personali. Abbiamo già visto come potete mostrare i processi, filtrare quelli che appartengono ad un determinato utente e quali possibili restrizioni si possono incontrare. Ci sono due cose che potete fare quando vi accorgete che uno dei vostri processi sta consumando troppe risorse di sistema:

1. fare in modo che il processo usi meno risorse senza interromperlo;
2. fermare il processo completamente.

Se desiderate che il processo continui, ma volete lasciare maggior spazio agli altri processi del sistema, potete usare **renice** sul medesimo. A parte l'uso dei comandi **nice** o **renice**, **top** è un modo facile per individuare i processi problematici e per ridurre la priorità.

Identificate il processo nella colonna "NI": molto probabilmente sarà quello con una priorità negativa. Battete **r** ed inserite l'ID del processo a cui volete modificare la priorità. Indicate poi il valore "nice" (per esempio "20"): ciò sta ad indicare che tale processo userà al massimo 1/5 dei cicli di CPU.

Esempi di processi che volete mantenere in funzione sono: gli emulatori, le macchine virtuali, i compilatori e così via.

Se desiderate fermare un processo perché si impianta o sta divenendo completamente un berserk sulla strada del consumo dell'I/O, nella creazione di file o nell'uso di altre risorse, utilizzate il comando **kill**. Se ne avete l'opportunità, prima provate a "uccidere" elegantemente il processo, inviando il segnale SIGTERM. Questa è un'istruzione per terminare qualsiasi cosa si stia facendo, secondo le procedure descritte nel codice del programma:

```
joe:~> ps -ef | grep mozilla
joe    25822 1 0 Mar11 ?       00:34:04 /usr/lib/mozilla-1.4.1/mozilla-

joe:~> kill -15 25822
```

Nell'esempio qui sopra, l'utente *joe* ha fermato il suo browser Mozilla perché questo si era bloccato.

Alcuni processi sono più difficili da eliminare. Se ne avete il tempo, potete inviare loro il segnale SIGINT per interromperli. Se ciò non funziona, usate il segnale più forte, SIGKILL. Nell'esempio seguente, *joe* ferma un Mozilla che si è bloccato:

```
joe:~> ps -ef | grep mozilla
joe    25915 1 0 Mar11 ?       00:15:06 /usr/lib/mozilla-1.4.1/mozilla-

joe:~> kill -9 25915

joe:~> ps -ef | grep 25915
joe     2634 32273 0 18:09 pts/4 00:00:00 grep 25915
```

In questi casi, potreste desiderare di controllare quale processo è realmente morto, usando il filtro **grep** sul PID. Se quest'ultimo restituisce solo il processo **grep**, potrete stare sicuri di essere riusciti nel fermarlo.

Fra i processi che sono ardui da uccidere c'è la vostra shell. E ciò è una buona cosa: se essa fosse semplice da uccidere, voi potreste perdere la shell ogni qualvolta premete accidentalmente **Ctrl-C** nella linea di comando, da momento che ciò equivale ad inviare un SIGINT.

> **Linux è quasi impensabile senza le "pipe"**
> L'uso delle "pipe" (|) per inviare i dati in uscita di un comando all'ingresso di un altro viene spiegato nel prossimo capitolo 5.

In un ambiente grafico, il programma **xkill** risulta piuttosto facile da impiegare: basta solo digitare il nome del comando, seguito da un **Invio**, e scegliere la finestra dell'applicazione da fermare. E' comunque anche piuttosto pericoloso perché invia normalmente un SIGKILL: usatelo perciò solo quando un'applicazione si impianta.

4.4. Temporizzare i processi.

4.4.1. Usate quel tempo di ozio!

Un sistema Linux può patire molto, ma solitamente soffre solo durante le ore d'ufficio. Che si trovino in un ambiente d'ufficio, in una stanza dei server o a casa, molti sistemi Linux oziano durante il primo mattino, la sera, la notte ed i fine settimana. L'utilizzo di tale tempo di inattività può essere molto più economico dell'acquisto di quelle macchine che dovreste comprare volendo fare tutto nel medesimo tempo.

Esistono tre tipi di esecuzione ritardata:

- attendere per un poco e poi riassumere l'esecuzione di un processo, usando il comando **sleep**. Il tempo di esecuzione dipende dal tempo del sistema al momento dell'invio.
- avviare un comando in un momento specificato, usando il comando **at**. L'esecuzione del processo (o di più processi) dipende dal tempo di sistema, non dal momento dell'invio, non da quello di invio.
- avviare un processo regolarmente con cadenza mensile, settimanale, giornaliera od oraria, usando i mezzi offerti da **cron**.

Le seguenti sezioni trattano di ciascuna possibilità.

4.4.2. Il comando sleep

La pagina Info dedicata a sleep è forse una delle più corte. Tutto ciò che fa sleep è attendere. Normalmente il tempo da aspettare viene espresso in secondi.

Allora, perché esiste? Alcuni esempi pratici:

Qualcuno telefona e voi rispondete “Si sarò lì da te tra mezz'ora”, ma siete veramente sommersi da lavoro e siete pure costretti a rinunciare al pranzo:

(sleep *1800*; echo “*Ora di pranzo...*”) &

Quando per qualche motivo non potete usare il comando **at** (sono le cinque), volete andare a casa ma c'è ancora lavoro da sbrigare e proprio adesso qualcuno sta sottraendo risorse del sistema:

(sleep *10000*; mioprogramma) &

Avviando questo genere di processo assicuratevi che ci sia una disconnessione automatica nel vostro sistema e di uscire o di bloccare il vostro ufficio/desktop, oppure avviatelo in una sessione di **screen**.

Quando lanciate una serie di stampe di grossi file, ma desiderate che altri utenti possano stampare nel frattempo:

lp `mucchioditesti`; sleep *900*; lp `grossofile`; sleep *900*; lp `altrogrossofile`

Nel Capitolo 8 tratteremo della stampa dei file.

I programmatori usano spesso il comando **sleep** per fermare l'esecuzione di script o programmi per un certo lasso di tempo.

4.4.3. Il comando at

Il comando **at** esegue dei comandi ad una data ora, utilizzando la vostra consueta shell a meno che non diate il comando in altro modo (v. pagina man).

```
steven@home:~> at tomorrow + 2 days
warning: commands will be executed using (in order) a) $SHELL
        b) login shell c) /bin/sh
at> cat reports | mail myboss@mycompany
at> <EOT>
job 1 at 2001-06-16 12:36
```

Battendo **Ctrl+D** si interrompe l'utility **at** e si genera il messaggio “EOT”.

L'utente *steven* qui fa una cosa strana combinando due comandi; studieremo questo tipo di esempio nel Capitolo 5 – Redirezione dell'Input e dell'Output.

```
steven@home:~> at 0237
warning: commands will be executed using (in order) a) $SHELL
        b) login shell c) /bin/sh
at> cd new-programs
at> ./configure; make
at> <EOT>
job 2 at 2001-06-14 02:00
```

L'opzione `-m` invia un messaggio di posta all'utente al termine del processo, oppure avvisa quando quest'ultimo non è stato completato. Il comando **atq** elenca i processi; date tale comando prima di lanciare dei processi in modo da evitare di farli partire contemporaneamente ad altri. Con il comando **atrm** potete rimuovere dei processi pianificati, se cambiate propositi.

Una buona idea è scegliere strani orari di esecuzione poiché i processi di sistema vengono spesso avviato allo scoccare delle ore, come potete constatare nella prossima Sezione 4.4.4. Per esempio i processi spesso vengono lanciati all'una precisa del mattino (l'indicizzazione del sistema per l'aggiornamento del database standard di locate), cosicché la fissazione di un tempo pari a 0100 potrebbe facilmente rallentare il vostro sistema piuttosto che farlo accelerare. Per impedire che i processi siano avviati tutti allo stesso tempo, potete usare anche il comando **batch**, che li accoda e li sottopone al sistema in un modo equilibrato allo scopo di prevenire picchi eccessivi di uso delle risorse di sistema. V. le pagine Info per ulteriori informazioni.

4.4.4. Cron e crontab

Il sistema cron viene gestito dal demone cron: riceve informazioni su quando e quali programmi avviare in base alle istruzioni del sistema e delle “crontab” degli utenti. Solo l'utente root ha accesso alle crontab di sistema, mentre ciascun utente accede esclusivamente alle proprie crontab. In alcuni sistemi gli utenti (alcuni) possono essere privi delle funzionalità di cron.

Durante l'avvio del sistema il demone cron cerca le istruzioni crontab in `/var/spool/cron` che sono elencate dopo gli account in `/etc/passwd`, in `/etc/cron.d` e in `/etc/crontab`, quindi usa queste informazioni ogni minuto per controllare se c'è qualcosa da fare. Esso esegue i comandi come l'utente che possiede il file crontab ed invia per posta ogni output dei comandi al medesimo.
Nei sistemi che utilizzano il Vixie cron, i processi che si ripetono con cadenza oraria, giornaliera, settimanale e mensile sono conservati in directory separate sotto `/etc` per vederli rapidamente, al

contrario della funzione cron standard UNIX dove tutti i compiti sono inseriti in un unico grosso file.

Esempio di un file Vixie crontab:

```
[root@blob /etc]# more crontab
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# run−parts
# commands to execute every hour
01 * * * * root run−parts /etc/cron.hourly
# commands to execute every day
02 4 * * * root run−parts /etc/cron.daily
# commands to execute every week
22 4 * * 0 root run−parts /etc/cron.weekly
commands to execute every month
42 4 1 * * root run−parts /etc/cron.monthly
```

Alternativa
Potreste usare anche il comando **crontab -** `l` per vedere i crontab.

Vengono impostate delle variabili e poi c'è la vera pianificazione, una linea per ogni processo con 5 campi iniziali per l'ora e il giorno. Il primo campo contiene i minuti (da 0 a 59), il secondo definisce l'ora di esecuzione (0-23), il terzo rappresenta il numero del giorno (1-31), il quarto il numero del mese (1-12) e l'ultimo è il giorno della settimana (0-7, dove sia 0 che 7 indicano la domenica). Un asterisco in questi campi rappresenta l'intera serie di valori accettati. Gli elenchi sono consentiti: inserirete nell'ultimo campo 1-5 per eseguire un processo dal lunedì al venerdì e 1,3,5 per l'esecuzione nei giorni lunedì, mercoledì e venerdì.

Dopo questi 5 campi viene l'utente che dovrebbe avviare i processi elencati nell'ultima colonna. L'esempio qui sopra è ricavato da una configurazione di Vixie cron in cui *root* avvia il programma **run-parts** ad intervalli regolari con le directory appropriate come opzioni. In queste directory i veri processi da eseguire all'orario pianificato sono conservati come script di shell, come questo piccolo script che viene avviato quotidianamente per aggiornare il database utilizzato dal comando **locate**:

```
billy@ahost cron.daily]$ cat slocate.cron
#!/bin/sh
renice +19 −p $$ >/dev/null 2>&1
/usr/bin/updatedb −f "nfs,smbfs,ncpfs,proc,devpts" −e \
"/tmp,/var/tmp, /usr/tmp,/afs,/net"
```

Si suppone che gli utenti modifichino i propri crontab in modo sicuro utilizzando il comando **crontab -e** : ciò impedirà l'apertura accidentale di più di una copia del file crontab. Solitamente l'editor è **vi** (v. Capitolo 6, però potete usare qualsiasi editor testuale, come **gvim** o **gedit,** se vi trovate meglio con un editor visuale).

Al termine il sistema vi comunicherà che è stato installato un nuovo crontab.

Il crontab seguente ricorda a *billy* di recarsi ogni giovedì sera al proprio club sportivo:

```
billy:~> crontab -l
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (/tmp/crontab.20264 installed on Sun Jul 20 22:35:14 2003)
# (Cron version -- $Id: chap4.xml,v 1.24 2006/10/26 15:37:52 tille Exp $)
38 16 * * 3 mail -s "sports evening" billy
```

Dopo aver aggiunto una nuova attività pianificata, il sistema vi dirà che è stato installato un nuovo crontab. Non è necessario riavviare il demone **cron** per rendere efficaci i cambiamenti. Nell'esempio *billy* ha aggiunto una nuova riga che punta ad uno script di backup:

```
billy:~> crontab -e
45 15 * * 3 mail -s "sports evening" billy
4 4 * * 4,7 /home/billy/bin/backup.sh

<--write and quit-->

crontab: installing new crontab

billy:~>
```

Lo script `backup.sh` viene eseguito ogni giovedì e domenica (v. Sezione 7.2.5 per un'introduzione agli script di shell). Tenete presente che, se presente, l'output dei vostri comandi viene inviato come e-mail al proprietario del file crontab. Se non fosse stato configurato alcun servizio di posta, potreste trovare l'output come semplice file di testo nella vostra casella postale locale `/var/spool/mail/<vostro_nomeutente>`.

Chi avvia i miei comandi?

Non dovete specificare l'utente che dovrebbe avviare i comandi in quanto vengono eseguiti normalmente con i permessi propri dell'utente.

4.5. Sommario

Linux è un sistema operativo multiutente, multitasking, che impiega un modo simile a UNIX per gestire i processi. La velocità di esecuzione dei comandi può dipendere da migliaia di piccoli fattori: fra l'altro, abbiamo imparato molti nuovi comandi per visualizzare e manipolare i processi. Eccone una lista:

Tabella 4-3. Nuovi comandi nel capitolo 4: Processi

Comando	Significato
at	Accoda i processi per la successiva esecuzione.
atq	Elenca i processi dell'utente in esecuzione.
atrm	Cancella processi in base al loro specifico numero di processo.
batch	Esegue comandi quando il livello di carico di sistema lo permette.
crontab	Gestisce i file crontab per ogni utente.
halt	Ferma il sistema.

Comando	Significato
init ***run level***	Elabora il processo di inizializzazione.
jobs	Elenca i processi correntemente in esecuzione.
kill	Termina un processo.
mesg	Controlla l'accesso in scrittura del vostro terminale.
netstat	Mostra le connessioni di rete, le tabelle di instradamento, le statistiche di interfaccia, le connessioni mascherate e le appartenenze del multicast.
nice	Avvia un programma con la priorità prevista modificata.
pgrep	Mostra i processi.
ps	Presenta lo stato dei processi.
pstree	Mostra un albero dei processi.
reboot	Riavvia il sistema.
renice	Modifica la priorità di esecuzione dei processi.
shutdown	Spegne il sistema.
sleep	Ritarda per un tempo determinato.
time	Temporizza un comando o mostra l'uso delle risorse.
top	Mostra i processi principali
uptime	Mostra da quanto tempo il sistema sta funzionando
.stat	Presenta le statistiche della memoria virtuale.
w	Mostra chi è connesso al sistema e che cosa sta facendo
wall	Invia un messaggio a tutti i terminali.
who	Mostra chi è connesso al sistema
write	Manda un messaggio ad un altro utente.

4.6. Esercizi

Questi sono alcuni esercizi che vi aiuteranno a comprendere il significato dei processi che girano nel vostro sistema.

4.6.1. In generale

- Avviate **top** in un terminale mentre eseguite gli esercizi in un altro.
- Avviate il comando **ps**.
- Leggete le pagine **man** per cercare il modo di visualizzare tutti i vostri processi

- Avviate il comando **find /**. Quale effetto produce sul carico di sistema? Fermate questo comando.
- In modalità grafica, avviate il programma **xclock** in primo piano e poi lasciatelo funzionare dietro le quinte. Fermate il programma usando il comando **kill**
- Avviate direttamente in background il programma **xcalc** in modo da liberare il prompt del terminale da cui è stato avviato.
- Cosa fa **kill -9 -1**?
- Aprite nuovamente due terminali o finestre di terminale e usate **write** per inviare un messaggio dall'una all'altra.
- Date il comando **mesg**. Che cosa dice?
- Quanto ci vuole per eseguire **ls** nella directory corrente?
- In base alle informazioni dei processi in `/proc` di vostra proprietà, come dovete fare per scoprire quali processi attualmente rappresentano?
- Da quanto tempo è attivo il vostro sistema?
- Qual'è la vostra attuale TTY?
- Indicate 3 processi che non possono aver avuto **init** come originario genitore.
- Elencate 3 comandi che usano la modalità SUID. Spiegate perché ciò avviene.
- Nominate i comandi che generalmente causano il maggior carico nel vostro sistema.

4.6.2. Avvio, init, ecc...

- Potete riavviare il sistema in qualità di normale utente? Perché è così?
- In base al vostro attuale livello di avvio, elencate i passaggi da effettuare per la chiusura.
- Come cambiate livello di avvio del sistema? Passate dal vostro livello di esecuzione standard al livello 1 e viceversa.
- Elencate tutti i servizi e demoni avviati dall'accensione del vostro sistema.
- Quale kernel viene caricato attualmente all'avvio?
- Fingete di dover avviare un qualche strano server al momento dell'accensione: finora, dopo l'accensione del sistema vi autenticavate e lanciavate questo server manualmente ricorrendo allo script `consegna_pizza` della vostra directory personale. Cosa dovete fare per consentire l'avvio automatico del servizio nel run level 4 da voi creato solo per tale scopo?

4.6.3. Pianificazione

- Usate **sleep** per creare un avviso che la vostra pasta è pronta in dieci minuti.
- Create un processo **at** che copi tutti i file della vostra directory personale in `/var/temp` dopo mezzora. Potreste creare anche una sottodirectory in `/var/temp`.
- Create un processo cron che esegua questo compito dal lunedì al venerdì all'ora di pranzo.
- Controllate che funzioni.
- Inserite un errore nei dati crontab come, ad esempio, un inesistente comando **copy** al posto di **cp**. Cosa succede con l'esecuzione del compito?

Capitolo 5. Redirezione dell'I/O

Questo capitolo fornisce maggiori informazioni sul potente meccanismo UNIX della redirezione dell'input, dell'output e degli errori. Gli argomenti comprendono:

- Standard input, output ed errori
- Operatori di redirezione
- Come usare l'output di un comando come input di un altro
- Come creare un file con l'output di un comando per una successiva consultazione
- Come accodare l'output di più comandi in un file
- Redirezione dell'input
- Gestione dei messaggi standard di errore
- Combinare la redirezione dei flussi di input, output e di errore
- Filtri dell'output

5.1. Semplici redirezioni

5.1.1. Cosa sono lo standard input e lo standard output?

Molti comandi Linux leggono l'input, come file o altro attributo del comando, e scrivono l'output. Di norma l'input viene dato da tastiera e l'output appare sul vostro schermo. La tastiera costituisce il vostro dispositivo di *standard input* (stdin) [ingresso standard] e lo schermo o una specifica finestra di terminale è invece il dispositivo di *standard output* (stdout) [uscita standard].

Comunque, dal momento che Linux è un sistema flessibile, queste impostazioni di base non devono essere applicate necessariamente. Ad esempio lo standard output di un server molto controllato in un grande ambiente può essere una stampante.

5.1.2. Gli operatori di redirezione

5.1.2.1. Redirezione dell'output con > e |

Qualche volta potreste voler salvare l'output di un comando in un file oppure applicare un altro comando sull'output di un comando. Questa operazione è nota come redirezione dell'output. La redirezione si fa utilizzando sia “>” (simbolo di maggiore), sia l'operatore “|” (*pipe* in inglese) che invia lo standard output di un comando ad un altro come standard input.

Come abbiamo visto in precedenza, il comando **cat** concatena i file e li invia tutti insieme allo standard output. Redirigendo questo output ad un file, esso verrà creato – o sovrascritto se già esistente (fate attenzione).

```
nancy:~> cat test1
```

```
some words

nancy:~> cat test2
some other words

nancy:~> cat test1 test2 > test3

nancy:~> cat test3
some words
some other words
```

Non sovrascrivete!
State attenti a non sovrascrivere file esistenti (magari importanti) quando redirigete l'uscita dei dati. Molte shell, compresa Bash, hanno la capacità predefinita di proteggervi da tale rischio: **noclobber** (consultate le pagine Info per maggiori informazioni). Con Bash dovreste aggiungere il comando **set -o *noclobber*** al vostro file di configurazione `.bashrc` per prevenire scritture accidentali di file.

Redirigere "niente" verso un file esistente equivale a svuotarlo:

```
nancy:~> ls -l list
-rw-rw-r--    1 nancy nancy 117 Apr 2 18:09 list

nancy:~> > list

nancy:~> ls -l list
-rw-rw-r--    1 nancy nancy   0 Apr 4 12:01 list
```

Questo processo è chiamato *troncamento*.

La stessa redirezione ad un file inesistente creerà un nuovo file vuoto con il nome indicato:

```
nancy:~> ls -l newlist
ls: newlist: No such file or directory

nancy:~> > newlist

nancy:~> ls -l newlist
-rw-rw-r-- 1 nancy    nancy         0 Apr 4 12:05 newlist
```

Il Capitolo 7 fornisce alcuni altri esempi dell'uso di questo tipo di redirezione.

Ecco alcuni esempi dell'utilizzo dell'incanalamento (piping) dei comandi.

Per trovare una parola in un certo testo, mostrate tutte le linee contenenti "parola1" ed escludete quelle che contengono anche "parola2":

grep *parola1* file | grep -v *parola2*

Per mostrare l'output di un elenco dei file di una directory una pagina alla volta:

ls - la | less

Per trovare un file in una directory:

ls -l | grep ***parte_del_nome_del_file***

5.1.2.2. Redirezione dell'input

In altre occasioni potreste volere che un file costituisse l'input di un comando che normalmente non accetta file come opzione. Questa redirezione dell'input si ottiene ricorrendo all'operatore “<” (minore).

Qui sotto c'è un esempio di come inviare un file a qualcuno utilizzando la redirezione dell'input.

```
andy:~> mail mike@somewhere.org < to_do
```

Se l'utente *mike* esiste nel sistema, non avete bisogno di scrivere l'intero indirizzo. Se invece volete raggiungere qualcuno su Internet, battete l'intero indirizzo come argomento di **mail**.

Tale esempio si legge con maggiore difficoltà rispetto al banale **cat file | mail** ***qualcuno*** ma è un modo molto più elegante per usare gli strumenti disponibili.

5.1.2.3. Combinare redirezioni

L'esempio seguente combina le redirezioni di input e output. Il file `testo.txt` viene prima controllato alla ricerca di errori e l'output viene poi rediretto in un file di registrazione degli errori:

spell <testo.txt> errori.log

Il comando seguente elenca tutti i comandi che potete dare per esaminare un altro file quando usate **less**:

```
mike:~> less --help | grep -i examine
  :e [file]      Examine a new file.
  :n          * Examine the (N-th) next file from the command line.
  :p          * Examine the (N-th) previous file from the command line.
  :x          * Examine the first (or N-th) file from the command line.
```

L'opzione **-i** viene utilizzata per ricerche non sensibili a maiuscolo/minuscolo (ricordate che i sistemi UNIX sono molto “case sensitive”).

Se desiderate salvare l'output di questo comando per futura memoria, redirigetelo ad un file:

```
mike:~> less --help | grep -i examine > esaminare-file-in-less

mike:~> cat esaminare-file-in-less
  :e [file]      Examine a new file.
  :n          * Examine the (N-th) next file from the command line.
  :p          * Examine the (N-th) previous file from the command line.
  :x          * Examine the first (or N-th) file from the command line.
```

Di fatto l'output di un comando può essere incanalato in un altro comando quante volte volete a condizione che normalmente tali comandi leggano i dati in ingresso dallo standard input e scrivano

nello standard output quelli in uscita. Talvolta essi non sono in grado di farlo, ma potrebbero esserci delle speciali opzioni che istruiscono questi comandi a comportarsi secondo le regole consuete. Leggete perciò la documentazione (pagine man e Info) dei comandi utilizzati nel caso otteniate degli errori.

Nuovamente, siate certi di non usare nomi di file esistenti di cui avete ancora bisogno: la redirezione dell'output a file esistenti cancellerà il loro contenuto.

5.1.2.4. L'operatore >>

Invece di sovrascrivere i dati dei file, potete anche aggiungere del testo ad un file esistente utilizzando in successione due segni di maggiore.

Esempio:

```
mike:~> cat listadesideri
piu' soldi
meno lavoro

mike:~> date >> listadesideri

mike:~> cat listadesideri
piu' soldi
meno lavoro
Thu Feb 28 20:23:07 CET 2002
```

Il comando **date** scriverebbe sull'ultima linea sullo schermo: adesso invece la aggiunge al file `listadesideri`.

5.2. Caratteristiche avanzate della redirezione

5.2.1. Uso dei descrittori di file

Esistono tre tipi di I/O e ciascuno ha il proprio identificatore, chiamato descrittore di file:

- standard input: 0
- standard output: 1
- standard error: 2

Nelle descrizioni seguenti, se il numero descrittore di file viene omesso e il primo carattere dell'operatore di redirezione è <, la redirezione si riferisce allo standard input (descrittore di file 0): se invece il primo carattere dell'operatore di redirezione è >, allora la redirezione si riferisce allo standard output (descrittore di file 1).

Alcuni esempi pratici vi chiariranno maggiormente:

ls > **`dirlist`** **2>&1**

redirigerà sia lo standard output che lo standard error al file `dirlist`, mentre il comando

ls 2>&1 > dirlist

invierà solo lo standard output a `dirlist`. Questa può essere una opzione utile per i programmatori.

Le cose qui si stanno facendo abbastanza complicate: non confondete l'uso della E commerciale (&) con quello in Sezione 4.1.2.1., dove essa è usata per lanciare un programma in sottofondo. Qui serve esclusivamente per indicare che il numero seguente non è un nome di file, bensì una locazione a cui punta il flusso dei dati. Notate pure che il segno di maggiore non dovrebbe essere separato con spazi dal numero del descrittore di file: se venisse separato, noi punteremmo nuovamente l'uscita dei dati ad un file. L'esempio successivo ve lo dimostra:

```
[nancy@asus /var/tmp]$ ls 2> tmp

[nancy@asus /var/tmp]$ ls -l tmp
-rw-rw-r-- 1 nancy nancy 0 Sept   7 12:58 tmp

[nancy@asus /var/tmp]$ ls 2 > tmp
ls: 2: No such file or directory
```

Il primo comando eseguito da *nancy* è corretto (anche se non vengono generati errori ed è vuoto il file a cui lo standard error viene rediretto). Il secondo comando si aspetta che `2` sia un nome di file, che in questo caso è inesistente, cosicché viene segnalato un errore.

Tutte queste caratteristiche sono spiegate in dettaglio nelle pagine Info di Bash.

5.2.2. Esempi

5.2.2.1. Analisi degli errori

Se il vostro processo genera molti errori, questo è un modo per esaminarli approfonditamente:

comando 2>&1 | less

Ciò si usa spesso quando si creano nuovi programmi utilizzando il comando **make**, come in:

```
andy:~/newsoft> make all 2>&1 | less
--output ommitted--
```

5.2.2.2. Separazione dello standard output dallo standard error

Costrutti come questi vengono spesso usati dai programmatori in modo che l'uscita dei dati sia mostrata in una finestra di terminale e gli errori in un'altra. Scoprite quale pseudo terminale state utilizzando fornendo prima il comando **tty**:

```
andy:~/newsoft> make all 2> /dev/pts/7
```

5.2.2.3. Scrittura in contemporanea di output e file

Potete utilizzare il comando **tee** per copiare l'input nello standard output e in uno o più file in uscita in una sola mossa. Utilizzando l'opzione `-a` di **tee** si ha come risultato l'aggiunta dell'input in coda a(i) file. Tale comando è utile se volete nello stesso tempo vedere e salvare i dati in uscita. Gli operatori > e >> non permettono lo svolgimento di entrambe le azioni simultaneamente.

Questo strumento viene di solito invocato tramite una pipe (|), come mostrato nell'esempio seguente:

```
mireille ~/test> date | tee file1 file2
Thu Jun 10 11:10:34 CEST 2004

mireille ~/test> cat file1
Thu Jun 10 11:10:34 CEST 2004

mireille ~/test> cat file2
Thu Jun 10 11:10:34 CEST 2004

mireille ~/test> uptime | tee –a file2
 11:10:51 up 21 days, 21:21, 57 users,  load average: 0.04, 0.16, 0.26

mireille ~/test> cat file2
Thu Jun 10 11:10:34 CEST 2004
 11:10:51 up 21 days, 21:21, 57 users,  load average: 0.04, 0.16, 0.26
```

5.3. Filtri

Quando un programma svolge operazioni sui dati in ingresso e scrive il risultato nello standard output, esso viene definito filtro. Uno degli impieghi più comuni dei filtri è quello di ristrutturare l'output. Qui di seguito tratteremo di una coppia dei filtri più importanti.

5.3.1. Di più su grep

Come abbiamo visto nella Sezione 3.3.3.4, **grep** esamina i dati in usciti linea per linea ricercando le coincidenze al modello: tutte le linee che lo contengono vengono stampate nello standard output. Tale comportamento può essere invertito utilizzando l'opzione `-v`.

Ecco alcuni esempi.

Supponete di voler conoscere quali file di una certa directory sono stati modificati in febbraio:

```
jenny:~> ls –la | grep Feb
```

Il comando **grep**, come molti altri, è sensibile alle maiuscole/minuscole: usate perciò l'opzione `-i` per non distinguere tra di esse. Sono disponibili molte estensioni GNU, come `--colour`, che serve ad evidenziare i termini ricercati nelle linee lunghe, e –`after-context`, che stampa il numero di linea dopo l'ultima che corrisponde. Usando l'opzione `-r` potete avviare un **grep** ricorsivo che cerca in tutte le sottodirectory delle directory trovate. Come al solito le opzioni

possono essere combinate.

Le espressioni regolari si possono utilizzare per dettagliare maggiormente le esatte corrispondenze di caratteri che desiderate ricercare in tutte le linee in ingresso. Il modo migliore di cominciare con le espressioni regolari è, ovviamente, quello di leggere la documentazione di **grep**. Un eccellente capitolo è ricompreso nella pagina **Info** ***grep***. Dal momento che tutto ciò potrebbe condurci troppo lontano sui dettagli delle espressioni regolari, è fortemente consigliabile iniziare da qui se volete saperne di più.

Esercitatevi un po' con **grep:** vale proprio la pena di dedicare del tempo a questo piuttosto elementare, ma molto potente, comando di filtraggio. Gli esercizi alla fine di questo capitolo vi aiuteranno ad iniziare (v. Sezione 5.5.).

5.3.2. Filtraggio dei dati in uscita

Il comando **sort** normalmente dispone le linee in ordine alfabetico:

```
thomas:~> cat persone-preferite | sort

Amico
Capo
Mamma
Nonna
Papa'
Zietta Emmy
```

Ma esistono molte altre cose che **sort** è capace di fare: ad esempio guardare le dimensioni dei file. Con tale comando il contenuto della directory viene ordinato partendo dal file più piccolo a quello più grande:

ls -la | sort -nk5

Vecchia sintassi di sort
Potreste ottenere lo stesso risultato con **ls -la | sort +4n**, ma questa è una forma sorpassata non conforme agli standard correnti.

Il comando **sort** viene anche impiegato in combinazione con il programma **uniq** (o **sort -u**) per ordinare i dati in uscita e filtrarne i doppioni:

```
thomas:~> cat itemlist
1
4
2
5
34
567
432
567
34
555

thomas:~> sort itemlist | uniq
```

```
1
2
34
4
432
5
555
567
```

5.4. Sommario

In questo capitolo abbiamo imparato come i comandi possono essere combinati tra di loro e come l'input di un programma può essere usato come output di un altro.

La redirezione di input/output è un'attività comune nelle macchine UNIX e Linux. Tale potente meccanismo permette l'uso flessibile dei mattoni di cui è costituito UNIX.

Le redirezioni comunemente più usate sono > e |. Fate rifermento all'Appendice C per una panoramica sui comandi di redirezione ed altri costrutti di shell.

Tabella 5-1. Nuovi comandi nel capitolo 5: Redirezione dell'I/O

Comando	Significato
date	Mostra le informazioni orarie e di data.
set	Configura le opzioni di shell.
sort	Dispone in ordine linee di testo.
uniq	Rimuove da un file ordinato le linee duplicate.

5.5. Esercizi

Questi esercizi offrono più esempi di come combinare i comandi. L'obbiettivo principale è di provare ad usare il tasto **Invio** il meno possibile.

Tutti gli esercizi sono stati creati utilizzando un ID di utente normale in modo da produrre alcuni errori. Durante l'esecuzione non dimenticate di leggere le pagine man!

- Usate il comando **cut** sui dati in uscita di un lungo elenco della directory per mostrare solo i permessi dei file. Poi reindirizzate questo output a **sort** e **uniq** per filtrare tutte le linee duplicate. Infine usate **wc** per contare i diversi tipi di permesso contenuti in questa directory.
- Mettete l'output di **date** in un file. Aggiungete i dati in uscita di **ls** a questo file. Inviate il file risultante alla vostra casella postale locale (non specificate alcun `<@dominio>`, basterà solo il nome utente). Usando Bash, vedrete un avviso di posta nuova in caso di successo.
- Elencate le periferiche in `/dev` che al momento sono utilizzate dal vostro UID.

Reindirizzate attraverso **less** per vedere le loro proprietà.

- Date i comandi seguenti in qualità di utente non privilegiato. Determinate gli standard input, output ed error di ciascun comando:
    - cat `fileinesistente`
    - file `/sbin/ifconfig`
    - grep *`root`* `/etc/passwd /etc/nofiles > grepresults`
    - /etc/init.d/sshd *`start`* `> /var/tmp/output`
    - /etc/init.d/crond *`start`* `> /var/tmp/output 2>&1`
    - Ora controllate i vostri risultati ridando i comandi e redirigendo lo standard output al file `/var/tmp/output` e lo standard error al file `/var/tmp/error`.
- Quanti processi sono attualmente attivi?
- Quanti file invisibili esistono nella vostra directory personale?
- Usate **locate** per trovare documentazione sul kernel.
- Scoprite quale file contiene la seguente riga:

```
root:x:0:0:root:/root:/bin/bash
```

  e quest'altra:

```
system:                    root
```

- Guardate cosa succede battendo questo comando:

```
> time; date >> time; cat < time
```

- Quale comando usereste per verificare quale script in `/etc/init.d` avvia un certo processo?

Capitolo 6. Gli editor di testo

In questo capitolo discuteremo l'importanza del padroneggiare un editor, in particolare **vi** iMproved.

Al termine del capitolo sarete in grado di:

- aprire e chiudere file in modalità testo
- modificare file
- cercare del testo
- annullare errori
- fondere file
- recuperare file persi
- trovare un programma o una suite per l'uso in ufficio

6.1. Editor di testo

6.1.1. Perché dovrei usare un editor?

E' molto importante essere capaci di usare almeno un editor in modalità testuale. Sapere come utilizzare un editor nel vostro sistema è il primo passo per l'indipendenza.

Avremo bisogno di padroneggiare un editor dal prossimo capitolo non appena ci imbatteremo nella necessità di modificare i file che influenzano il nostro ambiente. Come un utente esperto, potrete voler cominciare a scrivere script o libri, sviluppare siti web o nuovi programmi. Saper usare un editor migliorerà immensamente la vostra produttività così come le vostre capacità.

6.1.2. Quale editor dovrei usare?

La nostra attenzione è rivolta agli editor testuali, che possono essere impiegati anche in sistemi privi di ambiente grafico ed in finestre di terminale. Il vantaggio aggiuntivo di conoscere alla perfezione un editor testuale si riscontra nell'utilizzo su macchine remote, poiché il lavoro con esso migliora enormemente la velocità della rete, non essendo necessario trasferire l'intero ambiente grafico su di questa.

Esistono, come al solito, molti modi di affrontare la questione. Andiamo a vedere quali editor sono normalmente disponibili.

6.1.2.1. ed

L'editor **ed** è orientato alla linea e utilizzato per creare, mostrare, modificare e manipolare in altri modi i file testuali, sia in maniera interattiva sia nell'uso con gli script di shell.

ed è l'originale editor di testo delle macchine UNIX e perciò è ampiamente disponibile. Comunque per molti scopi è stato superato dagli editor a tutto schermo come **emacs** e **vi** (v. sotto).

6.1.2.2. GNU Emacs

Emacs è l'editor estensibile, configurabile, autodescrittivo, con schermo in tempo reale, noto in molti UNIX ed altri sistemi. Il testo da editare è visibile sul monitor e viene aggiornato molto spesso, normalmente dopo aver battuto uno o due caratteri. Ciò riduce al minimo la quantità di informazioni da tenere in mente mentre scrivete. Emacs è definito “avanzato” perché fornisce delle possibilità che vanno oltre i semplici inserimenti e cancellazioni: controllo dei sottoprocessi; indentazione automatica dei programmi; vista di due o più file alla volta; produzione di testo formattato e trattamento di caratteri, parole, linee, periodi, paragrafi e pagine così come di espressioni e commenti in molti linguaggi di programmazione differenti.

Autodescrittivo significa che in qualsiasi momento potete battere un carattere speciale, **Ctrl+H**, per scoprire quali opzioni avete a disposizione. Potete anche usarlo per scoprire cosa fa ogni comando o per trovare tutti i comandi che riguardano un determinato argomento.

Personalizzabile significa che potete modificare facilmente le definizioni dei comandi di Emacs. Per esempio, se utilizzate un linguaggio di programmazione i cui commenti iniziano con “<**” e finiscono con “**>”, potete istruire i comandi di manipolazione dei commenti in Emacs a usare quelle stringhe. Un altro genere di personalizzazione è la ridefinizione dell'insieme dei comandi: per esempio, se preferite che i quattro comandi fondamentali di movimento del cursore (su, giù, sinistra e destra) siano disposti a diamante sulla tastiera, potete riassegnare i tasti in quel modo.

Estensibile significa che potete andare oltre la semplice personalizzazione e scrivere per intero nuovi comandi, programmi nel linguaggio Lisp che gireranno sotto l'interprete Lisp proprio di Emacs. Emacs è un sistema estensibile *online*, in quanto è composto da molte funzioni che si chiamano reciprocamente, ciascuna delle quali può essere ridefinita nel mezzo di una sessione di scrittura. Quasi tutte le parti di Emacs possono essere sostituite senza necessità di fare una copia separata di tutto Emacs. Molti dei comandi di Emacs per la redazione del testo sono già scritti in Lisp: le poche eccezioni sono state scritte in C per efficienza, ma si sarebbero potute scrivere anche in Lisp. Sebbene solo un programmatore potrebbe scrivere un'estensione, tuttavia dopo chiunque può usarla.

Quando gira sotto il sistema a finestre X Emacs (avviato con **xemacs**) fornisce i propri menu e dei comode associazioni con i tasti del mouse. Ma Emacs può offrire molti dei vantaggi di un sistema a finestre in un terminale esclusivamente testuale. Per esempio, potete osservare o modificare alcuni file alla volta, spostare del testo tra file e redigere file mentre sono attivi dei comandi di shell.

6.1.2.3. Vi(m)

Vim sta per “Vi iMproved” [Vi migliorato]. Era nato come “Vi Imitation”, ma ci sono così tanti miglioramenti da giustificare il cambio del nome. Vim è un editor di testo che comprende quasi tutti

i comandi del programma UNIX **vi** e molti altri di nuovi.

I comandi nell'editor **vi** vengono inseriti solo da tastiera con il vantaggio di mantenere le dita su di essa e lo sguardo sullo schermo, piuttosto che portare il braccio ripetutamente al mouse. Per quelli che lo desiderano, si possono attivare sia il supporto per il mouse, sia una versione per la GUI con barre di scorrimento e menu.

Per la redazione di file ci riferiremo a **vi** o **vim** nel corso di questo libro, mentre siete naturalmente liberi di usare il vostro editor prediletto. Comunque raccomandiamo di impratichirvi almeno negli elementi fondamentali di **vi**, in quanto è l'editor testuale standard in quasi tutti i sistemi UNIX, mentre Emacs può costituire un pacchetto opzionale. Possono esistere piccole differenze tra diversi computer e terminali, ma la questione centrale è che se potete lavorare con **vi**, siete pure in grado di sopravvivere in qualsiasi sistema UNIX.

Oltre al comando **vim**, i pacchetti vIm possono anche offrire **gvim**, la versione Gnome di **vim**. Gli utenti principianti potrebbero trovarlo più semplice da utilizzare, poiché i menu offrono aiuti quando dimenticate o non sapete come svolgere un particolare compito di redazione usando i comandi standard di **vim**.

6.2. Impiego dell'editor Vim

6.2.1. Due modi

L'editor **vi** è uno strumento piuttosto potente e ricomprende un manuale molto esteso, che potete attivare utilizzando il comando **:help** all'avvio del programma (invece di usare **man** o **info,** che non forniscono molte informazioni). Qui tratteremo solo degli elementi base per farvi iniziare.

Quello che rende **vi** disorientante per un principiante è che può operare in due modalità: comando e inserimento. L'editor inizia sempre in modalità comando: i comandi vi spostano nel testo, ricercano, rimpiazzano, marcano blocchi ed eseguono altre attività di elaborazione, ed, infine, alcuni di essi convertono l'editor in modalità inserimento.

Ciò significa che ogni tasto può avere facilmente non uno solo, ma due significati: può rappresentare un comando per l'editor nella modalità corrispondente, oppure un carattere che volete in un testo quando si trova in modalità inserimento.

Pronuncia
Si pronuncia “vi-ai”.

6.2.2. Comandi di base

6.2.2.1. Muoversi attraverso il testo

Normalmente è possibile muoversi nel testo con i tasti freccia. Se così non fosse, provate:

- **h** per spostare il cursore a sinistra
- **l** per spostarlo a destra
- **k** per spostarlo verso l'alto
- **j** per spostarlo verso il basso

SHIFT-G sposta il cursore alla fine del documento.

6.2.2.2. Operazioni elementari

Questi sono alcuni frequenti comandi **vi**:

- **n dd** cancella **n** linee a partire dalla posizione corrente del cursore
- **n dw** cancella **n** parole alla destra del cursore
- **x** cancella il carattere su cui è posizionato il cursore
- **:n** spostamento alla linea **n** del file
- **:w** salva (scrive) il file
- **:q** uscita dall'editor
- **:q!** forzatura dell'uscita quando volete abbandonare senza salvare un file modificato
- **:wq** salva ed esce
- **:w nuovofile** salva il testo in `nuovofile`
- **:wq!** scavalca il permesso di sola lettura (se avete il permesso di scavalcare i permessi, usando l'account di *root*)
- **/unastringa** cerca la stringa nel file e posiziona il cursore sotto la prima coincidenza.
- **/** esegue la ricerca precedente muovendo il cursore alla coincidenza successiva
- **:/1,$s/parola/altraparola/g** rimpiazza `parola` con `altraparola` nell'intero testo
- **yy** copia un blocco di testo
- **n p** lo incolla **n** volte
- **:recover** ripristina un file dopo una interruzione improvvisa

6.2.2.3. Comandi che pongono l'editor in modalità inserimento

- **a** aggiunge: muove il cursore di una posizione alla destra prima di entrare in modalità inserimento
- **i** inserisce
- **o** inserisce una linea vuota sotto la posizione corrente del cursore spostando quest'ultimo lì

Premendo il tasto **Esc** si ritorna in modalità comando. Se non siete sicuri della modalità in cui vi trovate perché state usando una versione piuttosto vecchia di **vi** che non riporta il messaggio “INSERT”, battete **Esc** e sarete certi di ritornare in modalità comando. E' possibile che il sistema

emetta un piccolo segnale se vi trovate già in tale modalità quando premete **Esc**, suonando o dando un avviso visivo (un lampeggio sullo schermo): questo è un comportamento normale.

6.2.3. La maniera semplice

Al posto di leggere il testo (che è piuttosto noioso), potete utilizzare il vimtutor per apprendere i primi comandi di Vim. Si tratta di un tutorial di trenta minuti che insegna le funzioni elementari di Vim in otto semplici esercizi. Anche se non potete imparare tutto di **vim** in mezz'ora, la lezione è costruita in modo da descrivere abbastanza i comandi mettendovi in grado di utilizzare Vim come editor tuttofare.

In UNIX e MS Windows, se Vim è stato installato correttamente, potete avviarlo dalla shell o dalla linea di comando inserendo **vimtutor**. Ciò creerà una copia del file della lezione, in modo che potrete modificarlo senza il rischio di danneggiare l'originale. Esistono alcune versioni tradotte della lezione. Per sapere se c'è quella vostra, usate il codice di linguaggio a due lettere: per il francese dovrebbe essere **vimtutor *fr*** (se installato nel sistema).

6.3. Linux in ufficio

6.3.1. Storia

Per l'intera ultima decade il settore dell'ufficio è stato dominato tipicamente da MS Office e, ammettiamolo, i formati Microsoft Word, Excel e PowerPoint sono degli standard industriali con cui prima o poi avrete a che fare.

Questa situazione di monopolio di Microsoft è stata sentita come un grosso impedimento alla venuta di nuovi utenti di Linux, cosicché un gruppo di sviluppatori tedeschi diedero inizio al progetto StarOffice, che era (e lo è tuttora) mirato a creare un clone di MS Office. La loro società è stata acquistata da Sun Microsystems alla fine degli anni 1990, poco prima della versione 5.2. Sun continua lo sviluppo ma ha vietato l'accesso ai codici. Tuttavia lo sviluppo in base all'insieme originale dei codici sorgenti continua nella comunità dell'Open Source, la quale ha ridenominato il progetto OpenOffice. OpenOffice è ora disponibile per una varietà di piattaforme, comprendente MS Windows, Linux, MacOS e Solaris. C'è una schermata nella Sezione 1.3.2.

Quasi in contemporanea, è stata avviata una coppia di altri progetto abbastanza noti. Anche KOffice, la suite da ufficio che era popolare abitualmente tra gli utenti SuSE, costituisce una alternativa molto comune all'uso di MS Office. Come l'originale, questo clone incorpora un programma compatibile MS Word ed Excel, e molto altro.

Progetti minori si occupano dei singoli programmi della suite MS di esempio, come Abiword e MS Wordview per la compatibilità con i documenti MS Word e Gnumeric per vedere e creare fogli elettronici compatibili con Excel.

6.3.2. Suite e programmi

Le attuali distribuzioni normalmente ci giungono con tutti gli strumenti necessari. Dal momento che esse forniscono linee guida e indici di ricerca eccellenti nei menu di Aiuto, non le tratteremo in dettaglio. Come riferimento date un'occhiata alla documentazione di sistema o ai siti web dei progetti, come:

- http://www.openoffice.org
- http://www.koffice.org
- Freshmeat e Sourceforge per vari altri progetti

6.3.3. Note

6.3.3.1. Uso generale dei documenti d'ufficio

Provate a limitare l'uso dei documenti d'ufficio solo agli scopi per cui sono stati creati: l'ufficio.

Un esempio: molti utenti Linux impazziscono se inviate loro una email che dice più o meno così nel contenuto: "Ciao, devo dirti qualcosa, leggi l'allegato" e quest'ultimo si rivela essere un documento compatibile MS Word contenente "Ciao amico mio, come va il tuo nuovo lavoro: hai tempo per pranzare con me domani?". Un'altra pessima idea è l'allegato con la vostra firma, per esempio. Se intendete firmare messaggi e file, usate GPG, la Guard Privacy GNU compatibile PGP oppure i certificati SSL (Secure Socket Layer).

Questi utenti non sono seccati perché sono impossibilitati a leggere tali documenti o perché sono preoccupati che detti formati generino normalmente file molto ingombranti, ma piuttosto perché stanno usando MS Windows e, eventualmente, per il lavoro aggiuntivo causato dal dover avviare alcuni programmi in più.

6.3.3.2. File di configurazione di sistema e di utente

Nel prossimo capitolo cominceremo a configurare il nostro ambiente e ciò comporta la possibilità di modificare tutti i generi di file che determinano il comportamento di un programma.

Non create o modificate questi file con qualsiasi programma d'ufficio!

Le normali specifiche del formato del file potrebbero determinare che il programma aggiunga alcune linee di codice per definire il formato del file e i font utilizzati. Queste linee potrebbero essere male interpretate dai programmi che dipendono da esse, causando errori o il blocco del programma che legge il file. In alcuni casi potrete salvare il file come testo semplice, ma finirete nei guai se prenderete questa abitudine.

6.3.3.3. Ma io voglio un editor testuale grafico!

Se volete proprio insistere, provate **gedit**, **kedit**, **kwrite** o **xedit**: questi programmi creano solo file testuali, ciò di cui abbiamo bisogno. Se però progettate di scrivere qualcosa di serio, passate ad un

vero editor in modalità testuale come **vim** o **emacs**.

Un'alternativa accettabile è **gvim**, la versione Gnome di **vim**: dovrete usare ancora i comandi di **vi**, ma se vi troverete in difficoltà potrete cercarli nei menu.

6.4. Sommario

In questo capitolo abbiamo appreso l'uso di un editor. Mentre dipende dai vostri gusti personali quale usare, è necessario almeno sapere come si utilizza un editor.

L'editor **vi** è disponibile su ogni sistema UNIX.

Molte distribuzioni comprendono una suite da ufficio e un editor testuale grafico.

6.5. Esercizi

Questo capitolo ha un solo esercizio: avviate il Vim Tutor battendo **vimtutor** in una sessione di terminale ed iniziate.

In alternativa potreste avviare **emacs**, battere **Ctrl+H** e poi **T** per richiamare la lezione passo-passo di Emacs.

La pratica è l'unica via!

Capitolo 7. Home sweet /home

Questo capitolo è dedicato alla configurazione del vostro ambiente. Ora che sappiamo usare un editor, possiamo modificare tutti i tipi di file per farci sentire più a casa nostra. Dopo il completamento di questo capitolo ne saprete di più circa:

- l'organizzazione del vostro ambiente
- i file normali di configurazione della shell
- la configurazione della shell
- la configurazione del prompt
- la configurazione dell'ambiente grafico
- le applicazione sonore e video
- i gestori del video e delle finestre
- il funzionamento del sistema client-server di X
- le impostazioni della lingua e dei font
- l'installazione di nuovo software
- l'aggiornamento dei pacchetti esistenti

7.1. Corretta gestione della casa in generale

7.1.1. Introduzione

Come abbiamo menzionato in precedenza, è abbastanza semplice mettere a soqquadro il sistema. Non ci stancheremo mai abbastanza nell'insistere sull'importanza di mantenerlo in ordine. Se imparerete ciò sin dall'inizio, la cosa diventerà una sana abitudine che farà risparmiare tempo programmando con un sistema Linux o UNIX oppure nell'affrontare i compiti di gestione del sistema. Qui sono elencati alcuni modi per semplificarci la vita:

- Create una directory `bin` per i vostri file e script.
- Organizzate i file non eseguibili in directory appropriate e di queste createne a piacimento: per esempio create directory distinte per le immagini, i documenti, i progetti, i file scaricati, i fogli elettronici, i file personali, e così via.
- Rendete private le directory con il comando **chmod** ***`700`*** **`nomedirectory`**.
- Date ai vostri file nomi sensati, come `Lamentela al primo ministro 050302`, piuttosto di `lettera1`.

7.1.2. Fare spazio

In alcuni sistemi, **quota** può costringervi a fare pulizia di tanto in tanto, oppure i limiti fisici del vostro disco rigido potrebbero obbligarvi a recuperare spazio senza l'avvio di qualsiasi programma di monitoraggio. Questa sezione spiega un numero di modi, oltre all'uso del comando **rm**, per liberare spazio su disco.

Lanciate il comando **quota -v** per vedere quanto spazio è rimasto.

7.1.2.1. Svuotamento dei file

Qualche volta il contenuto di un file non vi interessa, però avete bisogno del nome del file come segno (per esempio vi serve solo l'ora di creazione di un file, un appunto che là un file c'era o ci potrebbe essere in futuro). Il modo per ottenere ciò nelle shell Bourne e Bash è quello di redirigere l'output di un comando nullo.

```
andy:~> cat desiderata > segnaposto

andy:~> ls -la segnaposto
-rw-rw-r--    1 andy   andy        200 Jun 12 13:34 segnaposto

andy:~> > segnaposto

andy:~> ls -la segnaposto

-rw-rw-r--    1 andy   andy          0 Jun 12 13:35 segnaposto
```

Il processo di riduzione di un file esistente ad uno con lo stesso nome che sia grande 0 byte si chiama *troncamento*.

Per creare un nuovo file vuoto lo stesso effetto si ottiene con il comando **touch**: con un file esistente **touch** aggiornerà soltanto l'ora di modifica. Leggete le pagine Info su **touch** per maggiori dettagli.

Per svuotare quasi completamente un file utilizzate il comando **tail**. Supponete che l'elenco dei *desiderata* dell'utente *andy* sia divenuto piuttosto lungo per le continue aggiunte in fondo senza la cancellazione delle cose già in possesso. Ora costui vuol mantenere solo gli ultimi cinque oggetti:

```
andy:~> tail -5 desiderata > nuovoelenco

andy:~> cat nuovoelenco > desiderata

andy:~> rm nuovoelenco
```

7.1.2.2. Approfondimento sui file di log

Alcuni programmi Linux insistono nello scrivere tutti i tipi di output in un file di registro (*log*). Solitamente esistono opzioni per registrare solo gli errori, oppure una quantità minima di informazioni (impostando per esempio il livello di scoperta degli errori del programma). Ma potrebbe anche non interessarvi un file di log. Ecco qui alcuni modi per sbarazzarsi di questi o almeno per limitarne le dimensioni:

- Provate a cancellare il file di registro quando il programma non è in funzione se siete certi di non averne ancora bisogno. Alcuni programmi possono anche verificare all'avvio se non esiste alcun file di log e dunque non registrare niente.
- Se cancellate il file di log e il programma lo ricrea, leggete la documentazione di quest'ultimo per cercare l'opzione che evita di scrivere file di registro.
- Tentate di creare file di log più piccoli registrando solo le informazioni per voi importanti o

solo significanti.

- Provate a sostituire il file di log con un collegamento simbolico a `/dev/null`; se sarete fortunati il programma non si lamenterà. Non fate ciò con i file di registro di programmi che vengono eseguiti con l'avvio del sistema o con i programmi lanciati da **cron** (v. Capitolo 4). Tali programmi potrebbero infatti sostituire il link simbolico con un piccolo file che inizierà nuovamente a crescere.

7.1.2.3. Posta

Fate pulizia con regolarità nella vostra casella postale, create sottocartelle e reindirizzate automaticamente usando **procmail** (leggete le pagine Info) o i filtri della vostra applicazione preferita per la lettura della posta. Se avete una cartella per i file da cestinare, svuotatela con regolarità.

Per reindirizzare la posta utilizzate il file `.forward` nella vostra directory personale. Il servizio di posta Linux cerca questo file ogni volta che deve consegnare della posta locale. Il contenuto del file determina il comportamento del sistema di posta nei confronti della vostra posta. Può avere una sola linea con un indirizzo di posta elettronica pienamente qualificato. In tal caso il sistema invierà tutta la vostra posta a quell'indirizzo. Per esempio, noleggiando dello spazio per un sito web potreste desiderare di reindirizzare la posta con destinatario il webmaster al vostro account personale per non sprecare spazio su disco. Il `.forward` del webmaster potrebbe apparire così:

```
webmaster@www ~/> cat .forward
mike@pandora.be
```

L'impiego dell'inoltro della posta è utile anche per evitarvi di dover controllare diverse caselle postali: potete infatti fare in modo che ogni indirizzo punti ad un account centralizzato e facilmente accessibile.

Potete anche domandare all'amministratore di sistema di definire un inoltro per voi nel file locale di alias di posta, come quando un account è stato chiuso ma la posta elettronica rimane attiva per un po' di tempo.

7.1.2.4. Risparmiare spazio con un link

Quando numerosi utenti devono accedere allo stesso file o programma, oppure quando il nome originale del file è troppo lungo o difficile da ricordare, usate un collegamento simbolico al posto di una copia distinta per ciascun utente o impiego.

Link simbolici multipli possono avere nomi differenti, come, ad esempio, un collegamento può essere chiamato `monfichier` nella directory di un utente e `miolink` in quella di un altro. Svariati collegamenti (con nomi diversi) al medesimo file possono coesistere nella stessa directory. Ciò capita frequentemente nella directory /lib: dando il comando

ls -l /lib

vedrete che questa directory è piena di link che puntano agli stessi file. Quest'ultimi vengono creati

affinché i programmi che ricercano un nome non restino bloccati ma vengano indirizzati al corretto/corrente nome delle librerie di cui hanno bisogno.

7.1.2.5. Limitare le dimensioni dei file

La shell contiene un comando interno per limitare le dimensioni dei file, **ulimit**, che può essere usato anche per mostrare le limitazioni nelle risorse di sistema:

```
cindy:~> ulimit -a
core file size (blocks)     0
data seg size (kbytes)      unlimited
file size (blocks)          unlimited
max locked memory (kbytes) unlimited
max memory size (kbytes)    unlimited
open files                  1024
pipe size (512 bytes)       8
stack size (kbytes)         8192
cpu time (seconds)          unlimited
max user processes          512
virtual memory (kbytes)     unlimited
```

Cindy non è una sviluppatrice e non si cura dei c.d. *core dump*, contenenti le informazioni di debug di un programma. Se vi servono i *core dump*, potete impostarne le dimensioni utilizzando il comando **ulimit**. Leggete le pagine Info dedicate a Bash per una spiegazione dettagliata.

File Core?
Un file core o *core dump* viene talvolta generato quando le cose vanno male con un programma durante la sua esecuzione. Il file core contiene una copia della memoria, così com'era al momento della verificazione dell'errore.

7.1.2.6. File compressi

I file compressi sono utili perché occupano meno spazio nel disco fisso. Un altro vantaggio è che si impegna meno larghezza di banda spedendo un file compresso in rete. Molti file, come le pagine man, vengono conservati in forma compressa dal sistema. D'altro canto, scompattarli per avere qualche informazione, per poi doverli nuovamente comprimere, è piuttosto oneroso in termini di tempo. Non volete, per esempio, espandere una pagina man, leggere di un'opzione di un comando e poi ricomprimere di nuovo la stessa pagina. Probabilmente molte persone si dimenticheranno di fare pulizia dopo aver trovato l'informazione di cui avevano bisogno.

Di conseguenza abbiamo degli strumenti che lavorano sui file compressi, scompattandoli solo in memoria. Il file compresso originario resta nel disco rigido così com'è. Molti sistemi supportano **zgrep**, **zcat**, **bzless** ed altri in modo da prevenire azioni di decompressione/compressione non necessarie. Osservate la vostra directory dei file binari di sistema e le pagine Info.

Guardate il Capitolo 9 per maggiori informazioni sulla effettiva compressione dei file e per esempi di creazione di archivi.

7.2. Il vostro ambiente testuale

7.2.1. Le variabili ambientali

7.2.1.1. In generale

Abbiamo già menzionato un paio di variabili d'ambiente, come `PATH` e `HOME`. Finora abbiamo visto soltanto esempi in cui esse servono alla shell per un certo scopo. Ma esistono molti programmi di utilità che hanno necessità delle informazioni su di voi per svolgere un buon servizio.

Di quali altre informazioni necessitano i programmi oltre a quelle sui percorsi e sulle directory personali?

Molti programmi vogliono conoscere il tipo di terminale che state adoperando; tale informazione viene conservata nella variabile `TERM`. In modalità testuale costituirà l'emulazione di terminale *linux*, in modalità grafica userete probabilmente *xterm*. Molti programmi desiderano conoscere qual'è il vostro editor preferito, nel caso che debbano avviare un editor in un sottoprocesso. La shell che state utilizzando viene registrata nella variabile `SHELL`, il tipo di sistema operativo in `OS` e così via. Un elenco di tutte le variabili correntemente definite per la vostra sessione può essere visualizzato inserendo il comando **printenv**.

Le variabili ambientali sono gestite dalla shell. Al contrario delle normali variabili di shell, quelle d'ambiente vengono ereditate da qualsiasi programma che avviate, compresa un'altra shell. Ai nuovi processi viene assegnata una copia di queste variabili, copia che essi potranno leggere, modificare e passare a loro volta ai propri processi figli.

Non c'è nulla di speciale nei nomi delle variabili se non che quelle comuni sono in lettere maiuscole per convenzione. Potete assegnare qualsiasi nome vogliate, sebbene esistano delle variabili standard abbastanza importanti da essere le stesse in ogni sistema Linux, come PATH e HOME.

7.2.1.2. Esportazione delle variabili

Il contenuto di una singola variabile normalmente viene mostrato usando il comando **echo**, come in questi esempi:

```
debby:~> echo $PATH
/usr/bin:/usr/sbin:/bin:/sbin:/usr/X11R6/bin:/usr/local/bin

debby:~> echo $MANPATH
/usr/man:/usr/share/man/:/usr/local/man:/usr/X11R6/man
```

Se volete modificare il contenuto di una variabile in un modo che sia utile agli agli altri programmi, dovete esportare il nuovo valore dal vostro ambiente a quello che sta eseguendo tali programmi. Un banale esempio è l'esportazione della variabile `PATH`. Potete dichiararla come segue per essere in grado di giocare con il programma di simulazione di volo che si trova in `/opt/FlightGear/bin`:

```
debby:~> PATH=$PATH:/opt/FlightGear/bin
```

Ciò istruisce la shell a non cercare i programmi soltanto nel percorso corrente ($PATH) ma anche nella directory aggiuntiva /opt/FlightGear/bin.

Comunque, finché il nuovo valore della variabile PATH non è noto all'ambiente, le cose non funzionano:

```
debby:~> runfgfs
bash: runfgfs: command not found
```

L'esportazione delle variabile si esegue usando il comando interno alla shell **export**:

```
debby:~> export PATH

debby:~> runfgfs
--flight simulator starts--
```

Con Bash normalmente questo si fa con un'unica mossa elegante:

export VARIABILE=*valore*

La stessa tecnica si utilizza per la variabile MANPATH, che dice al comando **man** dove cercare le pagine man compresse. Se viene aggiunto del software nuovo al sistema in directory nuove o strane, la sua documentazione si troverà probabilmente anch'essa in una inusuale directory. Se desiderate leggere le pagine man del nuovo software, estendete la variabile MANPATH:

```
debby:~> export MANPATH=$MANPATH:/opt/FlightGear/man

debby:~> echo $MANPATH
/usr/man:/usr/share/man:/usr/local/man:/usr/X11R6/man:/opt/FlightGear/man
```

Potete tralasciare la ribattitura di questo comando in ogni finestra che aprite aggiungendolo in uno dei vostri file di impostazione della shell (v. Sezione 7.2.2.).

7.2.1.3. Variabili riservate

La tabella seguente offre una panoramica delle variabili predefinite più comuni:

Tabella 7-1. Comuni variabili ambientali

Nome della variabile	Informazioni contenute
DISPLAY	usato dal sistema X Window per identificare il server di display
DOMAIN	nome di dominio
EDITOR	mantiene l'editor di linea favorito
HISTSIZE	misura in numero di linee del file history di shell
HOME	percorso della vostra directory personale

Nome della variabile	Informazioni contenute
HOSTNAME	nome host locale
INPUTRC	locazione del file di definizioni delle periferiche di input come la tastiera
LANG	lingua preferita
LD_LIBRARY_PATH	percorsi di ricerca delle librerie
LOGNAME	nome di login
MAIL	posizione della vostra cartella di posta in arrivo
MANPATH	percorsi di ricerca delle pagine man
OS	stringa che descrive il sistema operativo
OSTYPE	maggiori informazioni su versione, ecc...
PAGER	usata dai programmi come **man** che devono sapere cosa fare nel caso in cui l'output sia più di una finestra di terminale
PATH	percorsi di ricerca dei comandi
PS1	prompt primario
PS2	prompt secondario
PWD	Present Working Directory, ovvero la directory in cui ci si trova
SHELL	shell corrente
TERM	tipo di terminale
UID	ID utente (User ID)
USER(NAME)	nome utente
VISUAL	il vostro editor a tutto schermo preferito
XENVIRONMENT	posizione delle vostre impostazioni personali per il comportamento di X
XFILESEARCHPATH	percorsi di ricerca delle librerie grafiche

Molte variabili non solo sono predefinite ma anche preimpostate, utilizzando i file di configurazione. Discuteremo di ciò nella prossima sezione.

7.2.2. I file di impostazione della shell

Quando lanciate il comando **ls** `-al` per ottenere un lungo elenco di tutti i file, compresi quelli che iniziano con un punto, nella vostra directory personale, troverete uno o più file che iniziano con . e terminano con *rc*: nel caso di **bash** questo è `.bashrc`. Esso è la controparte del file globale di configurazione del sistema `/etc/bashrc`.

Autenticandosi in una shell di login interattiva, **login** compirà l'autenticazione, imposterà l'ambiente e avvierà la vostra shell. Nel caso di **bash** il passo successivo è la lettura del `profile` generale da

/etc, se esistente. Poi **bash** cercherà ~/bash_profile, ~/.bash_login e ~/.profile, in questa successione, e leggerà ed eseguirà i comandi dal primo che esiste ed è leggibile. Se non esiste, si applicherà /etc/bashrc.

Quando termina una shell di login, **bash** legge ed esegue i comandi dal file ~/.bash_logout, se esiste.

Questa procedura viene spiegata dettagliatamente nelle pagine man dedicate a **login** e a **bash**.

7.2.3. Un tipico insieme di file di configurazione.

7.2.3.1. Esempio di /etc/profile

Esaminiamo alcuni di questi file di configurazione. Per primo viene letto /etc/profile in cui vengono impostate importanti variabili come PATH, USER e HOSTNAME.

```
debby:~> cat /etc/profile
# /etc/profile

# System wide environment and startup programs, for login setup
# Functions and aliases go in /etc/bashrc


# Path manipulation
if [ `id –u` = 0 ] && ! echo $PATH | /bin/grep –q "/sbin" ; then
    PATH=/sbin:$PATH
fi


if [ `id –u` = 0 ] && ! echo $PATH | /bin/grep –q "/usr/sbin" ; then
    PATH=/usr/sbin:$PATH
fi


if [ `id –u` = 0 ] && ! echo $PATH | /bin/grep –q "/usr/local/sbin"
    then
    PATH=/usr/local/sbin:$PATH
fi


if ! echo $PATH | /bin/grep –q "/usr/X11R6/bin" ; then
    PATH="$PATH:/usr/X11R6/bin"
fi
```

Queste linee controllano il percorso da impostare: se *root* apre una shell (user ID 0), viene controllato che /sbin, /usr/sbin e /usr/local/sbin siano nel percorso. Se così non fosse, esse verrebbero aggiunte. Si controlla per tutti che /usr/X11R6/bin sia nel percorso.

```
# No core files by default
ulimit –S –c 0 > /dev/null 2>&1
```

Tutta la spazzatura finisce in /dev/null se l'utente non cambia questa impostazione.

```
USER=`id –un`
LOGNAME=$USER
MAIL="/var/spool/mail/$USER"
```

```
HOSTNAME=`/bin/hostname`
HISTSIZE=1000
```

Qui sono assegnati i valori appropriati alle variabili globali.

```
if [ -z "$INPUTRC" -a ! -f "$HOME/.inputrc" ]; then
    INPUTRC=/etc/inputrc
fi
```

Se la variabile `INPUTRC` non è impostata e non esiste `.inputrc` nella directory personale dell'utente, allora viene caricato il comune file di controllo dell'input.

```
export PATH USER LOGNAME MAIL HOSTNAME HISTSIZE INPUTRC
```

Vengono esportate tutte le variabili, cosicché esse sono a disposizione degli altri programmi che richiedono informazioni sul vostro ambiente.

7.2.3.2. La directory profile.d

```
for i in /etc/profile.d/*.sh ; do
    if [ -r $i ]; then
        . $i
    fi
done
unset i
```

Tutti gli script di shell leggibili dalla directory `/etc/profile.d` vengono letti ed eseguiti. Questi svolgono azioni come abilitare *color-ls*, creare l'alias **vi** a **vim**, impostare le variabili locali, ecc. La variabile temporanea `i` viene disattivata per impedire che disturbi il comportamento della shell più tardi.

7.2.3.3. Esempio di .bash_profile

Poi **bash** cerca `.bash_profile` nella directory personale dell'utente:

```
debby:~> cat .bash_profile
#################################################################
#                                                               #
#   .bash_profile file                                          #
#                                                               #
#   Executed from the bash shell when you log in.               #
#                                                               #
#################################################################

source ~/.bashrc
source ~/.bash_login
```

Questo file molto chiaro istruisce la vostra shell a leggere per prima cosa `~/.bashrc` e poi `~/.bash_login`. Troverete il comando interno alla shell **source** regolarmente quando lavorate in un ambiente di shell: si usa per applicare i cambiamenti della configurazione all'ambiente corrente.

7.2.3.4. Esempio di .bash_login

Il file `~/.bash_login` definisce la protezione base dei file impostando il valore di **umask** (v.

Sezione 3.4.2.2.). Il file `~/.bashrc` viene usato per stabilire un complesso di alias, funzioni e variabili d'ambiente personali peculiari all'utente. Per prima cosa legge `/etc/bashrc`, che descrive il prompt di base (PS1) e il valore normale di umask. Dopo di ciò potete aggiungere le vostre proprie impostazioni. Se non esiste `~/.bashrc`, normalmente viene letto `/etc/bashrc`.

7.2.3.5. Esempio di /etc/bashrc

Il vostro file `/etc/bashrc` potrebbe assomigliare a questo:

```
debby:~> cat /etc/bashrc
# /etc/bashrc

# System wide functions and aliases
# Environment stuff goes in /etc/profile

# by default, we want this to get set.
# Even for non-interactive, non-login shells.
if [ `id -gn` = `id -un` -a `id -u` -gt 99 ]; then
         umask 002
else
         umask 022
fi
```

Queste linee impostano il valore di **umask**. Poi, in base al tipo di shell, viene impostato l'invito (o *prompt*).

```
# are we an interactive shell?
if [ "$PS1" ]; then
  if [ -x /usr/bin/tput ]; then
    if [ "x`tput kbs`" != "x" ]; then
# We can't do this with "dumb" terminal
      stty erase `tput kbs`
    elif [ -x /usr/bin/wc ]; then
      if [ "`tput kbs|wc -c `" -gt 0 ]; then
# We can't do this with "dumb" terminal
         stty erase `tput kbs`
      fi
     fi
   fi
   case $TERM in
         xterm*)
         if [ -e /etc/sysconfig/bash-prompt-xterm ]; then
                 PROMPT_COMMAND=/etc/sysconfig/bash-prompt-xterm
         else
    PROMPT_COMMAND='echo -ne "\033]0;${USER}@${HOSTNAME%%.*}:\
${PWD/$HOME/~}\007"'
         fi
     ;;
         *)
    [ -e /etc/sysconfig/bash-prompt-default ] && PROMPT_COMMAND=\
/etc/sysconfig/bash-prompt-default
             ;;
     esac
     [ "$PS1" = "\\s-\\v\\\$ " ] && PS1="[\u@\h \W]\\$ "
     if [ "x$SHLVL" != "x1" ]; then # We're not a login shell
         for i in /etc/profile.d/*.sh; do
             if [ -x $i ]; then
                 . $i
             fi
         done
     fi
fi
```

7.2.3.6. Esempio di .bash_logout

Con la disconnessione vengono eseguiti i comandi in `~/.bash_logout`, i quali possono, per esempio, cancellare il terminale, in modo che abbiate una finestra pulita all'uscita di una sessione remota o nell'abbandonare la console di sistema:

```
debby:~> cat .bash_logout
# ~/.bash_logout

clear
```

Diamo un'occhiata più approfondita a come funzionano questi script nella prossima sezione. Tenete **info *bash*** a portata di mano.

7.2.4. Il prompt di Bash

7.2.4.1. Introduzione

Il prompt di Bash può fare molto di più che mostrare semplici informazioni come il vostro nome, il nome del vostro computer ed alcune indicazioni circa la directory correntemente in uso. Possiamo aggiungere altre informazioni come data e ora correnti, il numero degli utenti connessi, ecc...

Comunque, prima di cominciare, salveremo il nostro attuale prompt in un'altra variabile ambientale:

```
[jerry@nowhere jerry]$ MYPROMPT=$PS1

[jerry@nowhere jerry]$ echo $MYPROMPT
[\u@\h \W]\$

[jerry@nowhere jerry]$
```

Cambiando ora il prompt, dando per esempio il comando **`PS1="->"`**, potremo sempre ritornare a quello originario con il comando **`PS1=$MYPROMPT`**. Naturalmente, lo ritroverete anche alla riconnessione tutte le volte che vi gingillerete appena con il prompt nella linea di comando ed eviterete di inserirlo in un file di configurazione della shell.

7.2.4.2. Alcuni esempi

Per comprendere questi prompt e sequenze di *escape* usati, ci riportiamo alle pagine Info o man di Bash.

- **export `PS1="[\t \j] "`**
  Mostra l'ora del giorno ed il numero dei processi attivi.
- **export `PS1="[\d][\u@\h \w] : "`**
  Mostra data, nome utente, nome host e directory corrente di lavoro. Notate che \W mostra solo i nomi di base dell'attuale directory di lavoro.
- **export `PS1="{\!} "`**
  Mostra il numero cronologico (history) di ogni comando.
- **export `PS1="\[\033[1;35m\]\u@\h\[\033[0m\] "`**
  Mostra utente@host in rosa.

- export PS1="\[\033[1;35m\]\u\[\033[0m\] \[\033[1;34m\] \w\[\033[0m\] "
  Imposta il nome utente in rosa e la directory di lavoro corrente in blu.
- **export PS1="\[\033[1;44m\]$USER is in \w\[\033[0m\] "** Prompt per persone che hanno difficoltà nel distinguere la differenza tra il prompt stesso e ciò che essi scrivono.
- **export PS1="\[\033[4;34m\]\u@\h \w \[\033[0m\]"**
  Prompt sottolineato.
- **export PS1="\[\033[7;34m\]\u@\h \w \[\033[0m\] "**
  Caratteri in bianco su sfondo blu.
- export PS1="\[\033[3;35m\]\u@\h \w \[\033[0m\]\a"
  Prompt rosa in un carattere più chiaro che avvisa quando i vostri comandi sono terminati.
- **export PS1=...**

Le variabili vengono esportate in modo che anche i comandi eseguiti successivamente conoscano l'ambiente. La linea di configurazione del prompt preferito è meglio metterla nel file di configurazione `~/.bashrc` della vostra shell.

Se lo desiderate, i prompt possono eseguire script di shell e comportarsi diversamente a seconda delle diverse condizioni. Potete avere pure il prompt che suona una musica ogni volta che date un comando, sebbene ciò diviene quasi subito fastidioso. Maggiori informazioni si possono trovare nel Bash-Prompt HOWTO.

7.2.5. Gli script di shell

7.2.5.1. Cosa sono gli script?

Uno script di shell è, come abbiamo visto negli esempi sulla configurazione della shell, un file testuale contenente dei comandi della shell. Quando tale file viene utilizzato come primo argomento non di opzione nell'invocare Bash, e non sono state inserite né l'opzione `-c` né `-s`, Bash legge ed esegue i comandi presi dal file e poi esce. Questo modo di operare crea una shell non interattiva. Quando Bash esegue uno script di shell, viene attribuito il parametro speciale `0` al nome del file piuttosto che il nome della shell, e i parametri posizionali (qualsiasi cosa che segua il nome dello script) vengono fissati sui restanti argomenti, se esistenti. Se invece non sono stati forniti argomenti aggiuntivi, i parametri posizionali restano indefiniti.

Uno script di shell può essere reso eseguibile usando il comando **chmod** per accendere il bit di esecuzione. Quando Bash trova tale file mentre sta cercando in `PATH` un comando, apre una sottoshell per eseguirlo. In altri termini, eseguire

nomefile *ARGOMENTI*

equivale all'esecuzione di

bash *nomefile ARGOMENTI*

se “nomefile” è uno script di shell eseguibile. Tale sottoshell reinizializza se stessa in modo che il risultato è come se fosse stata invocata una nuova shell per interpretare lo script, con l'eccezione che le posizioni dei comandi registrate dalla genitrice (v. **hash** nelle pagine Info) vengono mantenute dalla figlia.

Molte versioni di UNIX rendono ciò parte del meccanismo di esecuzione di un comando del sistema operativo. Se la prima riga di uno script comincia con i due caratteri “#!”, il resto della riga specifica un interprete per il programma. Così potete indicare **bash**, **awk**, **perl** o qualche altro interprete o shell e scrivere il resto del file di script in quel linguaggio.

Gli argomenti per l'interprete consistono in un singolo argomento opzionale dopo il nome dell'interprete nella prima linea del file di script, seguito dal nome del file di script e poi dai rimanenti argomenti. Bash svolgerà questa azione nei sistemi operativi che non la gestiscono da loro stessi.

Spesso gli script di Bash iniziano con

```
#! /bin/bash
```

(supponendo che Bash sia stata installata in `/bin`), poiché ciò assicura che Bash verrà utilizzata per interpretare lo script, anche se eseguito sotto un'altra shell.

7.2.5.2. Alcuni semplici esempi

Uno script molto semplice consiste in un unico comando che saluta l'utente che lo esegue:

```
[jerry@nowhere ~] cat hello.sh
#!/bin/bash
echo "Ciao $USER"
```

Di fatto lo script è costituito da un solo comando, **echo**, che usa il *valore* ($) della variabile ambientale `USER` per stampare una frase personalizzata con il nome dell'utente che fornisce il comando.

Un altro “monolinea”, impiegato per mostrare gli utenti connessi:

```
#!/bin/bash
who | cut -d " " -f 1 | sort -u
```

Qui di seguito c'è uno script formato da un numero maggiore di linee che utilizzo per eseguire copie di sicurezza di tutti i file in una directory. Per prima cosa lo script crea un elenco di tutti i file presenti nella directory corrente e lo mette nella variabile LIST. Poi imposta il nome della copia per ciascun file ed infine copia il file. Per ogni file viene stampato un messaggio:

```
tille:~> cat bin/makebackupfiles.sh
#!/bin/bash
# make copies of all files in a directory
LIST=`ls`
for i in $LIST; do
```

```
        ORIG=$i
     DEST=$i.old
     cp $ORIG $DEST
     echo "copied $i"
done
```

Inserendo solo una linea come **mv** **** *.old*** non funziona, come noterete sperimentandola su un gruppo di file di prova. Un comando **echo** è stato aggiunto per mostrare dell'attività: esso torna normalmente utile quando uno script non funziona. Inseritene uno dopo ciascun passaggio dubbio e scoprirete l'errore in un attimo.

La directory /etc/rc.d/init.d contiene una quantità di esempi. Diamo un occhio a questo script che controlla il server immaginario ICanSeeYou:

```
#!/bin/sh
# description: ICanSeeYou allows you to see networked people

# process name: ICanSeeYou
# pidfile: /var/run/ICanSeeYou/ICanSeeYou.pid
# config: /etc/ICanSeeYou.cfg

# Source function library.
. /etc/rc.d/init.d/functions

# See how (with which arguments) we were called.
case "$1" in
        start)
                echo —n "Starting ICanSeeYou: "
                daemon ICanSeeYou
                echo
                touch /var/lock/subsys/ICanSeeYou
                ;;
        stop)
                echo —n "Shutting down ICanSeeYou: "
                killproc ICanSeeYou
                echo
                rm —f /var/lock/subsys/ICanSeeYou
                rm —f /var/run/ICanSeeYou/ICanSeeYou.pid
                ;;
        status)
                status ICanSeeYou
                ;;
        restart)
                $0 stop
                $0 start
                ;;
        *)
                echo "Usage: $0 {start|stop|restart|status}"
                exit 1
esac

exit 0
```

Per prima cosa con il comando . (punto) viene caricato un gruppo di funzioni di shell, usate da quasi tutti gli script di shell in `/etc/rc.d/init.d`. Dopo viene dato un comando **case** che definisce quattro modi diversi di esecuzione dello script: un esempio potrebbe essere **IcanSeeYou** **`start`**. La decisione su quale **case** eseguire dipende dalla lettura del primo argomento dello script con l'espressione *$1*.

Quando viene dato un input non corretto, viene eseguito il *case* di base, segnato con un asterisco,

con cui lo script restituisce un messaggio di errore. La lista dei **case** termina con l'istruzione **esac**. Nel case *start* il programma server viene avviato come demone e gli vengono assegnati un ID di processo e un lock. Nel case *stop* il processo server viene rintracciato e fermato con rimozione del lock e del PID. Opzioni, come quella `daemon`, e funzioni, come `killproc`, vengono definite nel file `/etc/rc.d/init.d/functions`. Questa impostazione è specifica delle distribuzioni usate in questo esempio. Gli initscript del vostro sistema potrebbero usare altre funzioni, definite in altri file, o nemmeno una.

In caso di successo lo script restituisce un codice d'uscita 0 al processo genitore.

Questo script è un bell'esempio di impiego delle funzioni, che rendono lo script più facile da leggere e il lavoro viene svolto più velocemente. Osservate che esse usano **sh** invece di **bash** per essere utili ad un insieme più esteso di sistemi. In un sistema Linux invocare **bash** come **sh** è la conseguenza della shell che gira in modalità compatibile POSIX.

Le pagine man di **bash** contengono maggiori informazioni circa le combinazioni di comandi, i cicli for e while e le espressioni regolari, come pure esempi. Un comprensibile corso di Bash, con esercizi, destinato ad amministratori di sistema e utenti avanzati, è disponibile su http://tille.garrels.be/training/bash/, dalla stessa autrice di questa guida “Introduzione a Linux”. La descrizione dettagliata delle caratteristiche ed applicazioni di Bash si trova nella guida di riferimento Advanced Bash Scripting.

7.3. L'ambiente grafico

7.3.1. Introduzione

L'utente medio potrebbe non preoccuparsi eccessivamente delle proprie impostazioni di login, ma Linux offre un'ampia varietà di appariscenti gestori di finestre e desktop da usare sotto X, l'ambiente grafico. L'impiego e la configurazione di gestori di finestre e desktop sono chiari e possono anche assomigliare agli ambienti standard MS Windows, Apple o UNIX CDE, sebbene molti utenti Linux preferiscano desktop più colorati e gestori di finestre più fantasiose. Qui non vogliamo trattare della configurazione specifica degli utenti. Semplicemente sperimentate e leggete la documentazione usando le funzioni interne di Aiuto fornite da questi gestori e così procederete bene.

Daremo comunque uno sguardo più attento al sistema che sta alla base.

7.3.2. Il sistema X Window

Il sistema X Window è un sistema a finestre trasparente alla rete che gira in un'ampia gamma di macchine per calcoli e grafica. I server di sistema X Window funzionano su computer con schermo a bitmap. Lo X server distribuisce l'input dell'utente ed accetta richieste uscenti da svariati programmi attraverso una varietà di canali di comunicazione interprocesso differenti. Sebbene il caso più comune per i programmi clienti sia quello di girare sulla stessa macchina del server, i

clienti si possono egualmente eseguire in modo trasparente da altre macchine (comprese macchine con architetture e sistemi operativi differenti). Impareremo a fare ciò nel Capitolo 10 dedicato alle reti ed alle applicazioni in remoto.

X supporta la sovrapposizione gerarchica di sottofinestre ed operazioni di testo e grafica, su schermi sia monocromatici che a colori. Il numero dei programmi clienti X che usano il server X è abbastanza ampio. Alcuni dei programmi forniti nella distribuzione essenziale di X Consortium comprendono:

- xterm: un emulatore di terminale
- twm: un gestore di finestre minimalista
- xdm: un gestore di schermi
- xconsole: un programma di redirezione della console
- **bitmap**: un editor di bitmap
- xauth, xhost e iceauth: programmi di controllo degli accessi
- xset, xmodmap e molti altri: programmi per impostare le preferenze dell'utente
- xclock: un orologio
- xlsfonts e altri: un visualizzatore di font, utilità per elencare informazioni circa i font, le finestre e gli schermi
- xfs: un server di font
- ...

Ci riportiamo nuovamente alle pagine man di questi comandi per dettagliate informazioni. Maggiori spiegazioni sulle funzioni disponibili possono essere ricercate nel manuale *Xlib – C language X Interface* che trovate nella vostra distribuzione, nelle specifiche di *X Window System Protocol* e nei vari manuali e testi degli X toolkit. la directory `/usr/share/doc` contiene riferimenti a tali documenti ed a molti altri ancora.

Molte altri programmi di utilità, gestori di finestre, giochi, toolkit e gadget sono inclusi come contributi degli utenti nella distribuzione di X Consortium, oppure sono disponibili su Internet mediante FTP anonimo. Buoni posti dove incominciare sono http://www.x.org e http://www.xfree.org.

Inoltre tutte le applicazioni grafiche, come i vostri browser, programmi di posta elettronica e di visualizzazione di immagini, strumenti per la produzione di suoni e così via, sono tutti clienti del vostro server X. Notate che nelle normali operazioni in modalità grafica i clienti X ed il server X girano sotto Linux nella stessa macchina.

7.3.2.1. Nomi dei display

Dal punto di vista dell'utente ogni server X ha un *nome di schermo* (*display name*) nella forma:

`nomehost:numerodisplay.numeroscreen`

Tale informazione viene utilizzata dall'applicazione per stabilire come dovrà connettersi al server X e che schermo dovrà usare normalmente (su display con molti schermi):

- nomehost: rappresenta il nome della macchina cliente a cui è connesso fisicamente il display. Se non viene indicato, verrà usata la modalità più efficiente di comunicare ad un server sulla stessa macchina.
- numerodisplay: la parola “display” viene abitualmente usata per riferirsi ad un gruppo di monitor che condividono tastiera e puntatore (mouse, tavoletta grafica, ecc...). Molte workstation tendono ad avere solo una tastiera e perciò un unico display. D'altro canto sistemi multiutente più grandi hanno spesso diversi display cosicché più di una persona alla volta può lavorare in grafica. Per evitare equivoci ad ogni display di una macchina viene assegnato un numero di display (iniziando dallo 0) quando il server X di quel display è stato avviato. Il numero di display deve essere sempre assegnato in un nome di display.
- numero di screen: alcuni display condividono una sola tastiera ed un unico mouse fra due o più monitor. Siccome ogni monitor ha il proprio set di finestre, a ciascuno screen viene attribuito un numero di screen (cominciando dallo 0) all'avvio del server X per il display corrispondente. Se non viene indicato un numero di screen, si userà lo screen 0.

Nei sistemi POSIX il nome del display di partenza viene conservato nella variabile ambientale DISPLAY. Tale variabile è impostata automaticamente dall'emulatore di terminale **xterm**. Comunque, quando vi autenticate in un'altra macchina di una rete, potete aver bisogno di impostare manualmente DISPLAY per puntare al vostro display (v. Sezione 10.4.3.2.).

Maggiori informazioni possono essere scovate nelle pagine man di X

7.3.2.2. I gestori di finestre e desktop

La rappresentazione delle finestre nello schermo è controllata da programmi speciali chiamati *gestori di finestre* (o *window manager*). Sebbene molti di questi gestori di finestre seguano le specifiche di geometria così come assegnate, altri potrebbero scegliere di ignorarle (per esempio, chiedendo all'utente di disegnare esplicitamente la regione della finestra sullo schermo mediante il puntatore).

Dal momento che i gestori di finestre sono normali (anche se complessi) programmi clienti, può essere costruito un tipo diverso di interfaccia utente. La distribuzione di X Consortium viene fornita di un gestore di finestre chiamato **twm**, sebbene molti utenti preferiscano qualcosa di più fantasioso, risorse di sistema permettendo. Sawfish e Enlightenment sono esempi popolari che consentono a ciascun utente di avere un desktop intonato all'umore ed allo stile.

Un gestore di desktop (*desktop manager*) consente di usare un gestore di finestre oppure un altro per definire in modo adeguato il vostro desktop grafico, con menu a barre, menu a tendine, messaggi informativi, un orologio, un gestore di programmi e di file, ecc... Fra i desktop manager più in voga troviamo Gnome e KDE, che girano entrambi in quasi tutte le distribuzioni Linux ed in molti altri sistemi UNIX.

Applicazioni KDE con Gnome/applicazioni Gnome con KDE
Non è necessario avviare il desktop in KDE per far girare delle applicazioni KDE. Se avete installato le librerie KDE (pacchetto kdelibs), potete eseguire queste applicazioni dai menu Gnome o farle partire da un terminale di Gnome.

Far girare applicazioni Gnome in ambiente KDE è un attimo più difficoltoso perché sotto Gnome non esiste un insieme unico di librerie di base. Comunque le dipendenze e gli altri pacchetti extra che potreste dover installare vi verranno segnalati al momento dell'avvio o dell'installazione di tali applicazioni.

7.3.3. Configurazione di un server X

La distribuzione X normalmente fornita con Linux, *Xfree86*, utilizza il file di configurazione `XF86Config` per le sue impostazioni iniziali. Questo file configura la vostra scheda video e si trova in molte posizioni, sebbene quella più frequente sia in `/etc/X11`.

Se vedete che il file `/etc/X11/XF86Config` è presente nel vostro sistema, una sua descrizione può essere rintracciata nelle pagine Info o man con `XF86Config`.

A causa delle licenze adottate da Xfree86, i sistemi più recenti normalmente sono forniti con la distribuzione X.Org del server e degli strumenti X. Qui il file principale di configurazione è `xorg.conf`, anche questo solitamente in `/etc/X11`. Il file consta di un certo numero di sezioni disposte in qualsiasi ordine. Queste contengono informazioni circa il vostro monitor, la scheda video, la configurazione dello schermo, la tastiera, ecc... Come utenti, non dovete preoccuparvi eccessivamente di cosa c'è in quel file, dal momento che ogni cosa viene normalmente impostata quando si installa il sistema.

Se dovete cambiare le impostazioni del server grafico, potete comunque mandare in esecuzione gli strumenti di configurazione o modificare i file di configurazione che determinano l'infrastruttura per l'uso del server Xfree86. Date uno sguardo alle pagine man per maggiori informazioni; la vostra distribuzione potrebbe avere i suoi propri strumenti. Dal momento che una configurazione errata potrebbe causare della spazzatura illeggibile in modalità grafica, sarebbe consigliabile fare una copia di backup del file di configurazione prima di provare a modificarlo, tanto per essere sicuri.

7.4. Specifiche impostazioni di regione

7.4.1. Configurazione della tastiera

La tastiera per le console testuali si configura con il comando **loadkeys**. Utilizzate invece i vostri strumenti locali di configurazione di X oppure modificate manualmente la sezione *keyboard* in `XF86Config` per configurare la disposizione in modalità grafica. `XkbdLayout` è uno dei comandi che potreste utilizzare:

```
    XkbdLayout       “us”
```

Questo è di base. Per adeguarlo alle vostre impostazioni locali, modificatelo sostituendo il valore tra le virgolette con uno qualsiasi dei nomi elencati nelle sottodirectory della vostra directory `keymaps`. Se non riuscite a trovare le mappature dei tasti, provate a far apparire nel vostro sistema

la loro posizione dando il comando

locate keymaps

E' possibile combinare le impostazioni degli schemi, come in questo esempio:

```
Xkblayout "us,ru"
```

Fate una copia di sicurezza del file `/etc/X11/XF86Config` prima di modificarlo! Per fare ciò avete bisogno dell'account di *root*.

Disconnettetevi e poi riconnettetevi per ricaricare le impostazioni di X.

La Gnome Keyboard Applet vi permette di passare in tempo reale tra gli schemi; non sono richiesti speciali permessi per l'uso di questo programma. KDE possiede uno strumento analogo per passare da una configurazione di tastiera all'altra.

7.4.2. I tipi di caratteri

Utilizzate lo strumento **setfont** per caricare i font in modalità testo. Molti sistemi hanno un file standard `inputrc` che rende possibili combinazioni di caratteri come quello francese "é" (metacaratteri). L'amministratore di sistema dovrebbe poi aggiungere la linea

```
export INPUTRC="/etc/inputrc"
```

al file `/etc/bash`.

7.4.3. Data e fusi orari

L'impostazione dell'informazione oraria viene di solito effettuata al momento dell'installazione. Successivamente può essere mantenuta aggiornata ricorrendo ad un cliente *NTP* (Network Time Protocol). Molti sistemi Linux normalmente avviano **ntpd**:

```
debby:~> ps –ef | grep ntpd
ntp      24678     1 0 2002 ? 00:00:33 ntpd –U ntp
```

Potete avviare manualmente **ntpdate** per impostare l'ora, a condizione che possiate raggiungere un server temporale. Il demone **ntpd** non dovrebbe funzionare quando regolate l'ora con **ntpdate**. Utilizzate un server temporale quale argomento del comando:

```
root@box:~# ntpdate 10.2.5.200
26 Oct 14:35:42 ntpdate[20364]: adjust time server 10.2.5.200 offset
 -0.008049 sec
```

Date uno sguardo al vostro manuale di sistema e alla documentazione allegata al pacchetto NTP. Molti gestori di desktop comprendono strumenti per la regolazione dell'ora di sistema, consentendovi di aver accesso all'account di amministratore di sistema.

Per impostare il fuso orario (*time zone*) corretto, potete servirvi dei comandi **tzconfig** o **timezone**. L'informazione sul fuso orario viene normalmente data durante l'installazione della vostra macchina. Molti sistemi hanno degli strumenti specifici della distribuzione per configurarlo: date uno sguardo alla vostra documentazione di sistema.

7.4.4. La lingua

Se preferite piuttosto ricevere i messaggi dal sistema in olandese o in francese (ndt. in italiano), dovete impostare le variabili ambientali `LANG` e `LANGUAGE`, impostando così il supporto per la localizzazione al linguaggio desiderato ed eventualmente ai font legati ai segni convenzionali di tale linguaggio.

In molti sistemi grafici di autenticazione (come **gdm** o **kdm**), avete la possibilità di configurare queste impostazioni linguistiche prima del login.

Notate che in molti sistemi l'impostazione tipo tende ad essere *en_US.UTF-8* in questo periodo. Ciò non costituisce un problema poiché i sistemi che ce l'hanno di base sono forniti anche di tutti i programmi che supportano tale codifica. Così **vi** può modificare tutti i file del vostro sistema, **cat** non si comporta in modo strano, ecc...

I problemi sorgono quando vi collegate ad un sistema più vecchio che non supporta questa codifica dei font, oppure quando aprite un file codificato UTF-8 in un sistema che supporta solo i tipi di carattere ad 1-byte. Il programma di utilità **recode** potrebbe fornirvi un buon servizio per convertire i file da un insieme di caratteri ad un altro. Leggete le pagine man per una panoramica sulle caratteristiche e sull'uso. Un'altra soluzione potrebbe essere quella di lavorare temporaneamente con
una diversa definizione di codifica impostando la variabile ambientale `LANG`:

```
debby:~> acroread /var/tmp/51434s.pdf
Warning: charset "UTF-8" not supported, using "ISO8859-1".
Aborted

debby:~> set | grep UTF
LANG=en_US.UTF-8

debby:~> export LANG=en_US

debby:~> acroread /var/tmp/51434s.pdf
<--new window opens-->
```

Fate riferimento al sito web di Mozilla per indicazioni su come ricevere Firefox nella vostra lingua. Il sito web OpenOffice.org ha informazioni sulla localizzazione della vostra suite OpenOffice.org.

7.4.5. Specifiche informazioni nazionali

L'elenco degli HOWTO contiene riferimenti circa le istruzioni di localizzazione, in bangla, bielorusso, cinese, esperanto, finnico, francofono, ebraico, ellenico, lituano, polacco, portoghese,

serbo, slovacco, sloveno, spagnolo, thai e turco.

7.5. Installare nuovo software

7.5.1. In generale

Dopo l'installazione di Linux, molte persone si stupiscono alla vista del computer funzionante e utilizzabile. Parecchie distribuzioni contengono un ampio supporto per le schede video e di rete, per monitor ed altre periferiche esterne, cosicché di solito non serve installare driver extra. Pure strumenti comuni come le suite da ufficio, i navigatori web, i clienti di posta elettronica e così via, sono compresi nelle maggiori distribuzioni maggiori. Anche così una prima installazione potrebbe non corrispondere alle vostre esigenze.

Se proprio non trovate ciò che vi serve, potrebbe essere che non sia stato installato nel vostro sistema. Potrebbe anche essere che voi abbiate il software ricercato, ma che non faccia ciò che si supponeva dovesse fare. Ricordatevi che Linux si muove rapido ed il software migliora su base giornaliera. Non gettate il vostro tempo nel ricercare problemi che potrebbero essere stati già risolti. Potete aggiornare il vostro sistema oppure aggiungergli pacchetti quando volete. Molto software arriva in pacchetti. Software extra potrebbe essere rintracciato nei vostri CD di installazione o su Internet. Il sito web della vostra distribuzione è un buon posto per cominciare a cercare programmi aggiuntivi e contiene istruzioni su come installarli nella vostra versione di Linux (v. Appendice A). Leggete sempre la documentazione unita al nuovo software e qualsiasi istruzione per l'installazione eventualmente contenuta nel pacchetto. Tutti i software hanno un file `README`, che vi invito molto calorosamente a leggere.

7.5.2. I formati dei pacchetti

7.5.2.1. I pacchetti RPM

7.5.2.1.1. Cos'è un RPM?

RPM, il RedHat Package Manager, è un potente gestore di pacchetti che potete usare per installarli, aggiornarli e rimuoverli. Vi consente di cercare pacchetti e di mantenere traccia dei file contenuti in ogni pacchetto. E' integrato un sistema cosicché siete in grado di verificare l'autenticità dei pacchetti scaricati da Internet. Gli utenti avanzati hanno la possibilità di costruire i propri pacchetti con RPM.

Un pacchetto RPM consiste in un archivio di file e metadati utilizzati per installare e cancellare i file d'archivio. I metadati includono script di aiuto, attributi dei file, e informazioni descrittive del pacchetto. I pacchetti sono di due tipi: binari (usati per incapsulare il software da installare) e dei sorgenti (che contengono il codice sorgente e le regole necessari a produrre pacchetti binari).

Molte altre distribuzioni supportano i pacchetti RPM, fra cui le popolari RedHat Enterprise Linux, Mandriva (prima Mandrake), Fedora Core e SuSE Linux. A parte i consigli della vostra distribuzione, vi converrà leggere **man *rpm***.

7.5.2.1.2. Esempi di RPM

Numerosi pacchetti vengono semplicemente installati con l'opzione di aggiornamento, `-U`, che il pacchetto sia già installato o meno. Il pacchetto RPM contiene una versione completa del programma che sovrascrive le versioni esistenti o installa come uno nuovo. L'uso tipico è il seguente:

rpm –Uvh /path/to/rpm–package(s)

L'opzione `-v` genera un output più descrittivo e `-h` fa in modo che **rpm** stampi una barra di avanzamento.

```
[root@jupiter tmp]# rpm –Uvh totem–0.99.5–1.fr.i386.rpm
Preparing...                ########################################### [100%]
   1:totem                  ########################################### [100%]
[root@jupiter tmp]#
```

I pacchetti del nuovo kernel vengono installati con l'opzione `-i`, che evita di sovrascrivere le versioni esistenti del pacchetto. In questo modo sarete ancora in grado di avviare il vostro sistema con il vecchio kernel se quello nuovo non funziona.

Potete pure usare **rpm** per controllare se un pacchetto è installato nel vostro sistema:

```
[david@jupiter ~] rpm –qa | grep vim
vim–minimal–6.1–29
vim–X11–6.1–29
vim–enhanced–6.1–29
vim–common–6.1–29
```

Oppure potete scoprire quale pacchetto contiene un certo file od eseguibile:

```
[david@jupiter ~] rpm –qf /etc/profile
setup–2.5.25–1

[david@jupiter ~] which cat
cat is /bin/cat

[david@jupiter ~] rpm –qf /bin/cat
coreutils–4.5.3–19
```

Notate che non vi serve l'accesso con i privilegi di amministratore per interrogare mediante **rpm** il database di RPM. Dovete essere *root* solo quando aggiungete, modificate o cancellate pacchetti.

Di seguito un ultimo esempio che dimostra come si disinstalla un pacchetto ricorrendo a **rpm**:

```
[root@jupiter root]# rpm –e totem
[root@jupiter root]#
```

Osservate che la disinstallazione di base non è quella "verbosa": è normale che non vediate molto di ciò che sta succedendo. In caso di dubbi, usate di nuovo **rpm -qa** per verificare che il pacchetto sia stato rimosso.

RPM può fare molto di più della coppia di funzioni elementari che abbiamo trattato in questa introduzione: RPM-HOWTO contiene ulteriori informazioni.

7.5.2.2. I pacchetti DEB (.deb)

7.5.2.2.1. Cosa sono i pacchetti Debian?

Questo formato di pacchetti è quello standard di Debian GNU/Linux, dove **dselect**, e, più comune al giorno d'oggi, **aptitude**, è lo strumento normale di gestione dei pacchetti. Viene usato per selezionare i pacchetti che volete installare o aggiornare, ma si avvierà anche durante l'installazione di un sistema Debian e vi aiuterà a definire il metodo di accesso da utilizzare, ad elencare i pacchetti ed a configurarli.

Il sito web di Debian contiene tutte le informazioni che vi servono, compreso un "dselect Documentation for Beginners".

Seguendo le più recenti novità, il formato dei pacchetti di Debian sta divenendo sempre più popolare. Al momento di questo scritto, lo usano 5 delle distribuzioni top10. Anche **apt-get** (v. Sezione 7.5.3.2.) sta diventando estremamente popolare pure presso i sistemi non-DEB.

7.5.2.2.2 Esempi con gli strumenti DEB

La verifica sull'installazione di un pacchetto si effettua usando il comando **dpkg**. Per esempio, se volete sapere che versione del software Gallery è installata nella vostra macchina:

```
nghtwsh@gorefest:~$ dpkg -l *gallery*
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Installed/Config-files/Unpacked/Failed-config/Half-installed
|/ Err?=(none)/Hold/Reinst-required/X=both-problems (Status,Err: uppercase=bad)
||/ Name           Version        Description
+++-==============-==============-============================================
ii gallery         1.5-1sarge2    a web-based photo album written in php
```

Il prefisso “ii” significa che il pacchetto è installato. Se doveste vedere “un” come prefisso, allora ciò significherebbe che il pacchetto è presente nell'elenco tenuto dal vostro computer, ma che non è installato.

La ricerca sull'appartenenza di un file a quale pacchetto si effettua usando `-S` con **dpkg**:

```
nghtwsh@gorefest:~$ dpkg -S /bin/cat
coreutils: /bin/cat
```

Maggiori informazioni si possono trovare nelle pagine Info dedicate a **dpkg**.

7.5.2.3. I pacchetti dei sorgenti

La maggior parte dei programmi di Linux è a sorgente aperto/libero, cosicché sono disponibili i pacchetti dei sorgenti di questi programmi. I pacchetti dei sorgenti sono necessari per compilare la vostra versione personale di un programma. I sorgenti di un programma si possono scaricare dal suo sito web, spesso come un "tarball" compresso (`programma-versione.tar.gz` o qualcosa di simile). Per le distribuzioni basate su RPM, il sorgente è spesso fornito in `programma-versione.src.rpm`. Debian e molte distribuzioni basate su di questa, offrono da loro stesse il sorgente corretto che si può ricavare usando **apt-get** ***`source`***.

Nel file README sono riportate le specifiche richieste, dipendenze e istruzioni di installazione. Probabilmente vi servirà un compilatore C, **gcc**. Questo compilatore C GNU viene incluso in molti sistemi Linux ed è stato portato su molte altre piattaforme.

7.5.3. Gestione ed aggiornamenti automatici dei pacchetti

7.5.3.1. Note generali

La prima cosa da fare dopo l'installazione di un sistema nuovo è quella di applicare gli aggiornamenti: il suggerimento si applica a tutti i sistemi operativi e Linux non è differente.

Gli aggiornamenti per molti sistemi Linux normalmente si possono trovare in un vicino sito che fa da mirror della vostra distribuzione. Elenchi dei siti che offrono tale servizio si possono trovare nel sito web della vostra distribuzione (v. Appendice A).

Gli aggiornamenti dovrebbero essere effettuati regolarmente (ogni giorno, se possibile), ma ogni due settimane potrebbe essere un ragionevole inizio. Realmente dovreste tentare di avere la versione più recente della vostra distribuzione, dal momento che Linux cambia costantemente. Come abbiamo accennato in precedenza, funzioni, miglioramenti e correzioni di errori nuovi vengono forniti a ritmo serrato e qualche volta vengono pure sistemati problemi di sicurezza.

La buona notizia è che la maggior parte delle distribuzioni Linux fornisce strumenti in modo che non dovrete quotidianamente aggiornare a mano decine di pacchetti. Le sezioni seguenti vi offriranno una panoramica dei “gestori dei gestori di pacchetti”. C'è molto di più su questo argomento, persino aggiornamenti regolari dei pacchetti sorgenti sono gestibili automaticamente; noi elencheremo soltanto i sistemi maggiormente noti. Fate sempre riferimento alla documentazione della vostra particolare distribuzione per le procedure consigliate.

7.5.3.2. APT

Advanced Package Tool è un sistema di gestione dei pacchetti del software. Lo strumento a linea di comando per maneggiare i pacchetti è **apt-get**, che arriva fornito di una eccellente pagina man che descrive come installarli, aggiornarli singolarmente o nell'intera distribuzione. APT trae le sue origini dalla distribuzione Debian GNU/Linux, di cui rappresenta il normale gestore di pacchetti Debian. APT è stato modificato per supportare anche i pacchetti RPM. I maggiori vantaggi di APT consistono nell'essere libero e flessibile nell'impiego: vi permetterà di predisporre sistemi simili a quelli specifici della distribuzione (e, in alcuni casi, commerciali) elencati nelle prossime sezioni.

Generalmente, usando per la prima volta **apt-get**, vi servirà un indice di tutti i pacchetti disponibili. Per ottenere ciò si usa il comando

apt-get *update*

Successivamente potrete usare **apt-get** per aggiornare il vostro sistema:

`apt-get` ***`upgrade`***

Effettuare spesso questa operazione è un modo semplice per mantenere il vostro sistema al passo con i tempi e perciò sicuro.

A prescindere da questo impiego generico, **apt-get** è anche molto veloce nell'installare singoli pacchetti. Funziona così:

```
[david@jupiter ~] su - -c "apt-get install xsnow"
Password:
Reading Package Lists... Done
Building Dependency Tree... Done
The following NEW packages will be installed:
  xsnow
0 packages upgraded, 1 newly installed, 0 removed and 3 not upgraded.
Need to get 33.6kB of archives.
After unpacking 104kB of additional disk space will be used.
Get:1 http://ayo.freshrpms.net redhat/9/i386/os xsnow 1.42-10 [33.6kB]
Fetched 33.6kB in 0s (106kB/s)
Executing RPM (-Uvh)...
Preparing...                ########################################### [100%]
   1:xsnow                  ########################################### [100%]
```

Osservate l'opzione `-c` nel comando **su**, che indica alla shell di root di eseguire solo questo comando e di ritornarne poi nell'ambiente dell'utente. In questa maniera non vi scorderete di chiudere l'account di root.

Se esistono delle dipendenze con altri pacchetti, **apt-get** scaricherà e installerà tali pacchetti di supporto.

Maggiori informazioni si possono trovare in APT-HOWTO.

7.5.3.3. I sistemi che utilizzano i pacchetti RPM

Update Agent, che originariamente supportava solo i pacchetti RPM di RedHat, è stato ora allargato ad un esteso insieme di software, compresi i **repository** non-RedHat. Questo strumento offre un sistema completo per aggiornare i pacchetti RPM nei sistemi RedHat o Fedora Core. Nella linea di comando battete **up2date** per aggiornare il vostro sistema. Nel desktop viene normalmente attivata una piccola icona che vi avvisa se ci sono aggiornamenti disponibili per il vostro sistema.

Yellowdog's Updater Modifier (**yum**) è un altro strumento che recentemente è divenuto più popolare. Si tratta di un programma interattivo ma di aggiornamento automatizzato per installare, aggiornare o rimuovere pacchetti RPM in un sistema. E' lo strumento di elezione nei sistemi Fedora.

In SuSE Linux tutto si fa con Yast, Yet Another Setup Tool, che supporta un'ampia varietà di compiti di amministrazione di sistema fra cui l'aggiornamento dei pacchetti RPM. A partire da SuSE Linux 7.1 potete effettuare l'aggiornamento tramite un'interfaccia web e YOU, Yast Online Update.

Mandrake Linux e Mandriva offrono gli strumenti chiamati URPMI, un insieme di programmi di interfaccia che rendono più semplice l'installazione di nuovo software per l'utente. Tali strumenti

mettono insieme RPMDrake e MandrakeUpdate per fornire ogni cosa che serve per una facile installazione e disinstallazione dei pacchetti software. MandrakeOnline offre una estesa varietà di servizi e può comunicare automaticamente agli amministratori quando si rendono disponibili degli aggiornamenti per un particolare sistema Mandrake. Guardate, tra gli altri, **man *urpmi*** per maggiori informazioni.

Anche le suite per desktop KDE e Gnome hanno le loro specifiche (e grafiche) versioni dei gestori di pacchetti.

7.5.4. Aggiornare il kernel

La maggior parte delle installazioni Linux sono valide se aggiornate periodicamente la vostra distribuzione. La procedura di aggiornamento installerà un nuovo kernel, quando necessario, ed effettuerà tutte le opportune modifiche al vostro sistema. Dovrete compilare o installare un nuovo kernel manualmente solo se avrete necessità di funzioni del kernel non supportate da quello base incluso nella vostra distribuzione Linux. Ogni volta che compilate il vostro kernel personale ottimizzato o usate un pacchetto del kernel precompilato, installatelo in coesistenza con quello vecchio finché non siete sicuri che ogni cosa funzioni secondo le aspettative.

Poi create un sistema a doppio avvio, che vi consentirà di scegliere quale kernel avviare, modificando il vostro file di configurazione del boot loader, `grub.conf`. Questo è un semplice esempio:

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making config changes.
# NOTICE: You have a /boot partition. This means that
#          all kernel and initrd paths are relative to /boot/, e.g.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/hde8
#          initrd /initrd-version.img
#boot=/dev/hde
default=0
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
title Red Hat Linux new (2.4.9-31)
        root (hd0,0)
        kernel /vmlinuz-2.4.9-31 ro root=/dev/hde8
        initrd /initrd-2.4.9-31.img
title old-kernel
        root (hd0,0)
        kernel /vmlinuz-2.4.9-21 ro root=/dev/hde8
        initrd /initrd-2.4.9-21.img
```

Dopo che il nuovo kernel è stato collaudato positivamente, potete rimuovere dal file di configurazione di GRUB le linee che riguardano quello vecchio, anche se sarebbe preferibile attendere un paio di giorni solo per essere sicuri.

7.5.5. Installare pacchetti extra dai CD di installazione

7.5.5.1. Montare un CD

Di base, tale operazione si effettua nello stesso modo in cui si installano manualmente i pacchetti,

salvo che dovete aggiungere il file system del CD a quello della vostra macchina per renderlo accessibile. Nella maggior parte dei sistemi questo sarà fatto automaticamente con l'inserimento di un CD nel lettore perché al momento dell'avvio è stato lanciato il demone **automount**. Se il vostro CD non viene reso disponibile in modo automatico, date il comando **mount** in una finestra di terminale. In base alla vostra effettiva configurazione di sistema, una riga simile a questa normalmente risolverà il problema:

mount `/dev/cdrom /mnt/cdrom`

In alcuni sistemi solo *root* può montare delle periferiche rimovibili; ciò dipende dalla configurazione.

Per consentirne l'uso in automatico, il lettore CD di solito ha una linea in `/etc/fstab`, il quale elenca i file system ed i loro punti di montaggio, costituenti il vostro albero del file system. Si tratta di una riga simile a questa:

```
[david@jupiter ~] grep cdrom /etc/fstab
/dev/cdrom      /mnt/cdrom      iso9660 noauto,owner,ro 0 0
```

Ciò indica che il sistema sarà in grado di comprendere il comando **mount /mnt/cdrom**. L'opzione `noauto` significa che qui i CD non vengono montati nel momento dell'avvio.

Potete anche provare a premere con il tasto di destra del mouse sull'icona del CD nel desktop per montare il CD stesso se il vostro gestore di file non lo fa per voi. Potete controllare se ha funzionato dando il comando **mount** senza argomenti:

```
[david@jupiter ~] mount | grep cdrom
/dev/cdrom on /mnt/cdrom type iso9660 (ro,nosuid,nodev)
```

7.5.5.2. Usare il CD

Dopo aver montato il CD, siete in grado di cambiare directory, di solito al punto di montaggio `/mnt/cdrom`, dove potete accedere al contenuto del CD-ROM. Utilizzate gli stessi comandi per muovervi tra file e directory come fareste con i file nel disco rigido.

7.5.5.3. Espellere il CD

Per far uscire il CD dal lettore dopo il suo utilizzo, il file system del CD dovrebbe non essere in uso. Pur trovandosi in una sola delle sottodirectory del punto di montaggio (nel nostro esempio /mnt/cdrom), sarà considerato come file system in uso, cosicché dovrete uscire da là. Per esempio, fatelo battete **cd** senza argomenti (vi riporterà nella vostra directory personale. Dopo di ciò potete sia usare il comando

umount `/mnt/cdrom`

oppure

eject `cdrom`

Drive bloccati

MAI forzare i drive. Il trucco della graffetta è una pessima idea perché, anche se viene espulso il CD, il vostro sistema continuerà a credere che il CD sia ancora al suo posto dal momento che non sono state seguite le procedure normali. E' assai probabile che dovrete riavviare per far tornare il sistema in uno stato consistente.

Se continuate ad avere messaggi di "periferica occupata", controllate per prima cosa che tutte le sessioni di shell abbiano rilasciato il file system del CD e che non ci siano più applicazioni grafiche che lo stiano usando ancora. In caso di dubbi, usate lo strumento **lsof** per rintracciare il processo che sta utilizzando la risorsa del CD.

7.6. Sommario

Quando tutto è al suo posto significa che già metà del lavoro è fatto.

Anche se tenere in ordine è importante, altrettanto importante è il sentirsi "a casa" nel proprio ambiente, sia grafico che testuale. L'ambiente di testo è controllato attraverso i file di impostazione della shell. L'ambiente grafico, invece, dipende in primo luogo dalla configurazione del server X, su cui numero altre applicazioni sono state costruite, come i gestori di finestre e dei desktop e le applicazioni grafiche, ciascuna con i propri file di configurazione. Dovreste leggere la documentazione specifica del sistema e dei programmi per scoprire come configurarli.

Le impostazioni di regionalizzazione, come quella di tastiera, l'installazione di font appropriati e del supporto della lingua è meglio effettuarli al momento dell'installazione.

Il software viene gestito sia automaticamente che manualmente utilizzando un sistema di pacchetti.

Tabella 7-2. Nuovi comandi nel capitolo 7: Fare da voi stessi a casa

Comando	Significato
aptitude	Gestisce i pacchetti in stile Debian.
automount	Include automaticamente i file system appena inseriti.
dpkg	Il gestore Debian di pacchetti.
dselect	Gestisce i pacchetti in stile Debian.
loadkeys	Carica la configurazione della tastiera.
lsof	Identifica i processi.
mount	Include un nuovo file system nell'albero del file system esistente.
ntpdate	Imposta ora e data di sistema utilizzando un server temporale.
quota	Mostra informazioni circa l'uso dello spazio su disco concesso.
recode	Converte i file in un altro insieme di caratteri.

Comando	Significato
rpm	Gestisce i pacchetti RPM.
setfont	Sceglie un font.
timezone	Imposta il fuso orario.
tzconfig	Imposta il fuso orario.
ulimit	Imposta o mostra i limiti delle risorse.
up2date	Gestisce i pacchetti RPM.
urpmi	Gestisce i pacchetti RPM.
yum	Gestisce i pacchetti RPM.

7.7. Esercizi

7.7.1. L'ambiente della shell

- Stampate le impostazioni dell'ambiente. Quale variabile si può usare per memorizzare il tipo di CPU del vostro computer?
- Create uno script che possa scrivere qualcosa del tipo "ciao, mondo". Dategli i permessi appropriati in modo che possa essere eseguito. Provate il vostro script.
- Create una directory nella vostra directory personale e spostate lo script nella nuova directory. Aggiungete permanentemente questa nuova directory al vostro percorso di ricerca. Provate se lo script può essere eseguito senza dare il percorso alla sua reale posizione.
- Create delle sottodirectory nella vostra directory personale per conservare svariati file, per esempio una directory musica per registrare i file audio, una directory documenti per le vostre annotazioni, e così via. E usatele!
- Create un prompt personalizzato.
- Visualizzate i limiti dell'uso delle risorse. Riuscite a modificarli?
- Provate a leggere le pagine man compresse senza prima decomprimerle.
- Create un alias **lll** che in realtà esegua **ls `-la`**.
- Perché il comando **tail `testfile>testfile`** non funziona?
- Montate un CD dati, come ad esempio il vostro CD dell'installazione di Linux, ed esploratelo. Non scordatevi di smontarlo quando non vi serve più.
- Lo script della Sezione 7.2.5.2. non è perfetto: genera errori con i file che sono directory. Modificate lo script on modo che selezioni per la copia solo file normali. Usate **find** per effettuare la scelta. Non dimenticatevi di rendere il file eseguibile prima di provare ad avviarlo.

7.7.2. L'ambiente grafico

- Provate tutti i tasti del mouse in zone diverse (terminale, sfondo, barra delle funzioni).

- Esplorate i menu.
- Personalizzate la vostra finestra di terminale.
- Usate i bottoni del mouse per copiare e incollare del testo da un terminale ad un altro.
- Scoprite come configurare il vostro gestore di finestre. Provate diversi spazi di lavoro (schermi virtuali).
- Aggiungere una applet, come un analizzatore dei carichi, alla barra delle funzioni.
- Applicate un tema differente.
- Abilitate il cosiddetto *sloppy focus*, che si ha quando una finestra viene attivata passandoci sopra il mouse, cosicché non vi serve cliccare la finestra per abilitarla all'uso.
- Selezionare un gestore di finestre differente.
- Disconnettetevi e selezionate un tipo di sessione diverso, tipo KDE se prima stavate usando Gnome. Ripetete i passi precedenti.

Capitolo 8. Stampanti e stampe

In questo capitolo impareremo di più sulle stampanti e sui file di stampa. Dopo la lettura di questa parte sarete in grado di:

- Impostare il formato dei documenti
- Vedere in anteprima i documenti prima di inviarli alla stampante
- Scegliere una buona stampante che lavori con il vostro sistema Linux
- Stampare file e controllare lo stato della stampante
- Risolvere i problemi di stampa
- Trovare la documentazione necessaria per installare una stampante

8.1. I file di stampa

8.1.1. Stampare da linea di comando

8.1.1.1. Inviare il file alla stampante

La stampa da un'applicazione è molto semplice, selezionando l'opzione Stampa [o Print] dal menu.

Dalla linea di comando usate i comandi **lp** o **lpr**.

lp `file(s)`

lpr `file(s)`

Questi comandi possono leggere da un incanalamento [pipe], cosicché potete stampare l'output di comandi utilizzando

comando | lp

Esistono molte opzioni per regolare l'impostazione della pagina, il numero delle copie, la stampante con cui volete stampare se ne avete più di una, la dimensione della carta, la stampa normale o quella fronte-retro se la vostra stampante la consente, i margini e così via. Leggete le pagine man per una panoramica completa.

8.1.1.2. Lo stato dei vostri processi di stampa

Una volta che il file è stato accettato nella coda di stampa, viene attribuito un numero identificativo del processo di stampa:

```
davy:~> lp /etc/profile
request id is davy@blob+253
```

Per vedere (query) la coda di stampa, utilizzate i comandi **lpq** o **lpstat**. Se inseriti senza argomenti, mostrano i contenuti della coda di stampa preimpostata.

```
davy:~> lpq
blob is ready and printing
Rank     Owner  Job     File(s)         Total Size
active davy     253     profile         1024 bytes
davy:~> lpstat
blob–253        davy    1024    Tue 25 Jul 2006 10:20_01 AM CEST
```

8.1.1.3. Stato della vostra stampante

Qual'è la stampante predefinita in un sistema che accede a più stampanti?

lpstat -d

```
davy:~> lpstat -d
system default destination: blob
```

Qual'è lo stato della/e mia/e stampante/i?

lpstat -p

```
davy:~> lpstat -d
printer blob now printing blob–253. enabled since Jan 01 18:01
```

8.1.1.4. La rimozione dei processi dalla coda di stampa

Se non vi piace ciò che vedete con i comandi di stato, usate **lprm** o **cancel** per cancellare i processi.

```
davy:~> lprm 253
```

In ambiente grafico potete vedere apparire una finestra che vi comunica la cancellazione del processo.

In ambiti maggiori, **lpc** può essere utilizzato per controllare molteplici stampanti. Guardate le Info di ciascun comando.

Esistono molti strumenti di stampa ad interfaccia grafica (GUI) utilizzati come front-end di **lp** e la maggior parte delle applicazioni grafiche hanno una funzione di stampa che ricorre a **lp**. Leggete le funzioni interne di aiuto e la documentazione specifica dei programmi per maggiori informazioni.

Perché esistono due comandi per ogni operazione connessa alla stampa?
La stampa con UNIX e simili ha una lunga storia. Erano consueti due approcci piuttosto differenti: lo stile BSD di stampa e quello SystemV. Per compatibilità, Linux con CUPS supporta entrambi gli stili. Notate pure che **lp** non si comporta esattamente come **lpr**, **lpq** ha qualche opzione diversa da **lpstat** e **lprm** è quasi, ma non del tutto, simile a **cancel**. Non importa quali usate, basta che adoperiate quelli con cui vi ritrovate o che potreste conoscere per precedenti esperienze con sistemi simil-UNIX.

8.1.2. Impostazione dei formati

8.1.2.1. Strumenti e linguaggi

Se vogliamo ottenere qualcosa di sensato dalla stampante, i file devono prima assumere un formato. Linux, a prescindere dall'abbondanza di software per formattare, è fornito degli strumenti UNIX basilari per l'impostazione dei formati e per le lingue.

I sistemi Linux moderni supportano la stampa diretta, senza alcuna formattazione da parte dell'utente, di un ampio spettro di tipi di file: testo, PDF, PostScript e diversi formati di immagine come PNG, JPEG, BMP e GIF.

Per quei formati di file che necessitano di impostazioni, Linux dispone in quantità di strumenti di formattazione, come ad esempio i comandi **pdf2p**, **fax2ps** e **a2ps**, che convertono gli altri formati in PostScript. Questi programmi sono in grado di creare dei file che poi possono essere utilizzati in altri sistemi che non hanno installati tutti gli strumenti di conversione.

Esclusi questi strumenti a linea di comando, esistono parecchi programmi grafici di elaborazione testi. Sono disponibili diverse suite da ufficio complete, molte sono gratuite. Questi svolgono la formattazione in modo automatico al momento dell'invio di un processo di stampa. Solo per citarne alcuni: OpenOffice.org, Koffice, AbiWord, WordPerfect, ecc...

I seguenti sono linguaggi comuni in un contesto di stampa:

- **groff**: versione GNU del comando UNIX **roff**. Si tratta di un front-end del sistema groff di impaginazione dei documenti. Normalmente lancia il comando **troff** e un post-processore adatto alla periferica selezionata. Permette la generazione di file PotScript.
- *TeX* e il pacchetto di macro *LaTeX*: uno dei più largamente usati linguaggi di marcatura per i sistemi UNIX. Solitamente avviato con **tex**, imposta il formato dei file e restituisce in uscita la relativa rappresentazione (indipendente dalle periferiche) del documento di composizione.

  Le opere tecniche sono frequentemente scritte ancora in LaTeX perché supporta le formule matematiche, sebbene siano stati compiuti degli sforzi da parte del W3C (il World Wide Web Council) per includere tale funzionalità in altre applicazioni.
- SGML e XML: sono degli analizzatori liberi disponibili per UNIX e Linux. XML è SGML della prossima generazione, sta alla base del DocBook XML, un sistema di documenti (per esempio, questo libro è scritto in XML).

Documentazione sulla stampa
Le pagine man contengono dei dati preformattati di **troff** che devono essere impaginati prima di farli uscire dalla vostra stampante. La stampa si effettua usando l'opzione `-t` del comando **man**:

man -t *comando*>comando-man.ps

Successivamente si stampa il file PostScript. Se è stata configurata una destinazione predefinita di stampa per il vostro sistema/account, potete scrivere unicamente **man -t *comando*** per inviare direttamente la pagina formattata alla stampante.

8.1.2.2. Anteprima dei file impaginati

Qualsiasi cosa possiate inviare alla stampante, essa può altrettanto essere inviata allo schermo. In base al formato del file, potete usare uno di questi comandi:

- File PostScript: **gv** (GhostView).
- File TeX: **xdvi** o **kdvi** di KDE.
- File PDF: **xpdf**, **kpdf**, **gpdf** o **acroread**, il visualizzatore Adobe che è disponibile gratuitamente ma non è software libero. Il lettore Adobe supporta PDF 1.6 ,mentre gli altri solo le versioni fino ad 1.5. La versione di un file PDF può essere stabilita con il comando **file**.
- Nelle applicazioni, come Firefox o OpenOffice, normalmente potete selezionare l'anteprima di Stampa da uno dei menu.

8.2. Il lato server

8.2.1. In generale

Fino ad un paio di anni fa, la scelta per gli utenti Linux era semplice: ognuno lanciava lo stesso vecchio LPD, derivato dal codice di Net-2 di BSD. In seguito divenne molto popolare LPRng, ma al giorno d'oggi molte distribuzioni Linux moderne utilizzano CUPS, il Common UNIX Printing System. CUPS è un'implementazione dell'Internet Printing Protocol (IPP), un protocollo simil-HTTP standard RFC che rimpiazza il venerando (e difettoso) protocollo LDP. CUPS viene distribuito sotto GNU Public License ed è pure il sistema di stampa predefinito in MacOS X.

8.2.2. Configurazione grafica della stampante

La maggior parte delle distribuzioni è fornita di una GUI per la configurazione delle stampanti in rete e locali (porta parallela o USB). Vi permettono di scegliere il tipo di stampante da una lista e di provarla con facilità. Non dovete preoccuparvi della sintassi e della posizione dei file di configurazione. Controllate la documentazione del vostro sistema prima di tentare l'installazione della stampante.

CUPS può anche essere configurato usando un'interfaccia web attiva alla porta 631 del vostro computer. Per verificare se questa funzionalità è abilitata, provate a impostare la navigazione internet su localhost:631/help o localhost:631/.

8.2.3. L'acquisto di una stampante per Linux

Siccome sempre più produttori di stampanti mettono a disposizione driver per CUPS, quest'ultimo consentirà una facile connessione di qualsiasi stampante collegabile ad una porta seriale, parallela o USB, oltre a qualsiasi stampante in rete. CUPS vi permetterà una rappresentazione uniforme di tutti i diversi tipi di stampanti per voi e le vostre applicazioni.

Le stampanti dotate solamente un driver Win9x potrebbero creare dei problemi se non hanno altro supporto. In caso di dubbi verificate su http://linuxprinting.org/.

In passato la scelta migliore per voi avrebbe dovuto essere una stampante con supporto nativo di PostScript nel firmware, dal momento che quasi tutto il software UNIX e Linux che è in grado di produrre in uscita dati stampabili, lo fa in Postscript, il linguaggio d'elezione dell'industria editoriale per il controllo delle stampanti. Le stampanti PostScript sono di solito un po' più costose ma sono dotate di questo linguaggio di programmazione aperto, indipendente dalle periferiche, e perciò sarete sempre certi al 100% che funzioneranno. Tuttavia ai nostri giorni l'importanza di questa regola empirica sta scemando.

8.3. Problemi di stampa

In questa sezione tratteremo di ciò che potete fare da utenti quando qualcosa va nel verso sbagliato. Non vogliamo discutere qualsiasi problema che si può incontrare con la parte-demone del servizio di stampa, dal momento che si tratta di un compito da amministratore di sistema.

8.3.1. File sbagliato

Se stampate il file sbagliato, potete cancellare il processo usando il comando **lprm** ***`IDprocesso`***, dove *`IDprocesso`* è nella forma *nomestampante-numeroprocessodistampa* (ricavabile dalle informazioni mostrate da **lpq** o **lpstat**). Ciò funzionerà quando altri processi sono in attesa di essere stampati in questa coda di stampa. Comunque dovete essere veramente rapidi se siete gli unici ad usare questa stampante, dal momento che i processi sono normalmente immagazzinati in spool ed inviati alla stampante in pochi secondi. Una volta che essi abbiano raggiunto la stampante, è troppo tardi per rimuoverli usando gli strumenti Linux.

Ciò che potete tentare in quei casi, oppure nei casi in cui è configurato un driver sbagliato ed esce solo spazzatura dalla vostra stampante, è di spegnere la stampante. Tuttavia potrebbe non essere l'azione migliore in quanto potreste causare inceppamenti della carta ed altre irregolarità.

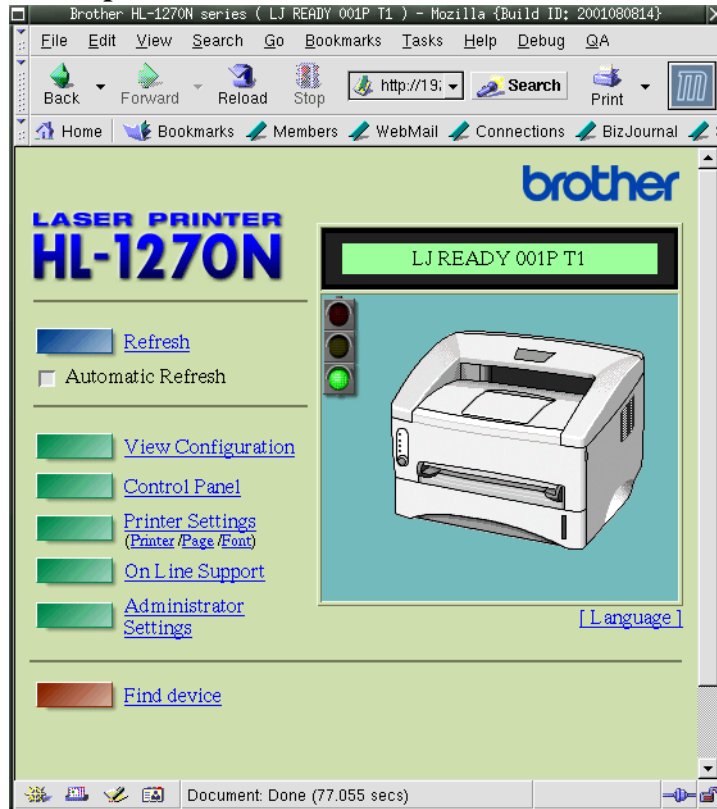
8.3.2. La mia stampa non è riuscita

Usate il comando **lpq** e cercate di individuare il vostro processo:

```
elly:~> lpq
Printer: lp@blob
  Queue: 2 printable jobs
  Server: pid 29998 active
  Unspooler: pid 29999 active
```

```
  Status: waiting for subserver to exit at 09:43:20.699
  Rank   Owner/ID             Class Job Files          Size Time
1       elly@blob+997           A   997 (STDIN)         129 09:42:54
2       elly@blob+22            A    22 /etc/profile     917 09:43:20
```

Molte stampanti ai nostri giorni hanno interfacce web che sono in grado di mostrare le informazioni di stato battendo l'indirizzo IP nel vostro navigatore web:

Figura 8-1. Stato della stampante attraverso l'interfaccia web

L'interfaccia web di CUPS rispetto all'interfaccia web della stampante

Fate attenzione che questa non è l'interfaccia web e funziona solamente con quelle stampanti che supportano tale funzionalità. Verificate la documentazione della vostra stampante.

Se l'ID del vostro processo non è là e neppure sulla stampante, contattate il vostro amministratore di sistema. Se questo ID invece è riportato nell'output, controllate che la stampante sia attualmente attiva: se così è, aspettate soltanto, il vostro processo verrà eseguito nei tempi dovuti.

Se la stampante non stampa, verificate che abbia carta, controllate le connessioni fisiche sia all'alimentazione elettrica che alla rete dei dati. Se ciò è a posto, la stampante potrebbe aver bisogno di essere riavviata. Chiedete un consiglio al vostro amministratore di sistema.

Nel caso di una stampante di rete, provate a stampare da un altro host. Se la stampante è

raggiungibile dal vostro host personale (guardate il Capitolo 10 per il programma di utilità **ping**), potete tentare di mettere il file impaginato su di essa, come `file.ps` in caso di stampante PostScript, usando un cliente FTP. Se ciò funziona, la vostra stampante è configurata male. Se invece non funziona, potrebbe essere che la stampante non comprenda il formato che le avete inviato.

Il sito GNU/Linux Printing contiene più consigli e trucchi.

8.4. Sommario

Il servizio Linux di stampa è dotato di un complesso di strumenti per la stampa basati su quelli standard UNIX LPD, che si tratti di un'implementazione SystemV o BSD. Sotto c'è un elenco dei comandi relativi alla stampa.

Tabella 8-1. Comandi nuovi nel capitolo 8: Stampa

Comando	Significato
lpr o **lp**	Stampa file
lpq o **lpstat**	Interroga la coda di stampa
lprm o **cancel**	Rimuove processi di stampa
acroread	Visualizzatore PDF
groff	Strumento di impaginazione
gv	Visualizzatore PostScript
printconf	Configura stampanti
xdvi	Visualizzatore DVI
xpdf	Visualizzatore PDF
***2ps**	Converte file in PostScript

8.5. Esercizi

Configurare e testare stampanti richiede la disponibilità di una almeno e di avere accesso all'account di *root*. Se così è, potete provare a:

- installare la stampante usando la GUI [Interfaccia Grafica per l'Utente] del vostro sistema;
- stampare una pagina di prova usando la GUI;
- stampare una pagina di prova con il comando **lp**;
- stampare da un'applicazione, per esempio Mozilla o OpenOffice, scegliendo File->Stampa dal menu;
- scollegare la stampante dalla rete o dal computer locale/server di stampa. Cosa succede se provate a stampare qualcosa?

Gli esercizi seguenti possono essere svolti senza stampante o accesso di root.

- Provate a creare file PostScript da differenti file sorgenti (per esempio HTML, PDF, pagine man). controllate i risultati con il visualizzatore **gv**.
- Verificate che il demone di stampa stia funzionando.
- Stampate i file in qualsiasi maniera. Cosa accade?
- Create un file Postscript usando Mozilla. Controllatelo con **gv**.
- Convertitelo in formato PDF. Controllatelo con **xpdf**.
- Come potreste avviare una stampa di un file GIF dalla linea di comando?
- Usate **a2ps** per stampare il file `/etc/profile` su un file in uscita. Verificatelo di nuovo con **gv**. Cosa succede se non specificate un file in uscita?

Capitolo 9. Tecniche fondamentali di backup

Incidenti prima o poi capiteranno. In questo capitolo tratteremo di come conservare i dati in un luogo sicuro usando altri host, dischetti, CD-ROM e nastri. Discuteremo inoltre dei comandi più popolari di compressione e archiviazione.

Al termine di questo capitolo saprete come:

- creare, ricercare e scompattare archivi di file;
- gestire dischetti e creare un disco di avvio per il vostro sistema;
- scrivere CD-ROM;
- effettuare copie di sicurezza incrementali;
- creare archivi Java;
- trovare documentazione per utilizzare altre periferiche di backup ed altri programmi;
- crittografare i vostri dati.

9.1. Introduzione

Sebbene Linux sia uno dei più sicuri sistemi operativi esistenti e pur essendo stato progettato per funzionare sempre, i dati possono andare persi. La perdita dei dati è molto spesso la conseguenza di errori dell'utente, ma occasionalmente un difetto del sistema (come un guasto elettrico o del disco) ne è la causa, cosicché è sempre una buona idea mantenere una copia extra di dati sensibili e/o importanti.

9.1.1. Preparazione dei vostri dati

9.1.1.1. Archiviare con tar

Nella maggioranza dei casi prima raccoglieremo tutti i dati da conservare in un unico file d'archivio, che poi compatteremo. Il processo di archiviazione implica il concatenamento di tutti i file elencati e l'estrazione di tutti gli spazi non necessari. In Linux di solito questo si fa con il comando **tar**. In origine **tar** era stato creato per archiviare dati su nastro, ma può anche creare archivi, noti come *tarball*.

tar ha molte opzioni, di cui citiamo qui sotto le più importanti:

- `-v`: dettagliata o “verbosa”
- `-t`: prova o test. Mostra il contenuto di una tarball
- `-x`: estrazione archivio
- `-c`: creazione archivio
- `-f perifericaarchivio`: usa `perifericaarchivio` come sorgente/destinazione della tarball. La periferica predefinita è la prima a nastro (di solito `/dev/st0` o qualcosa di simile)
- `-j`: filtra attraverso **bzip2** (guardate la Sezione 9.1.1.2.).

E' comune trascurare il prefisso meno [-] con le opzioni di **tar**, come potete vedere negli esempi seguenti.

Usate GNU tar per la compatibilità
Gli archivi creati con una versione proprietaria di **tar** in un sistema potrebbero essere incompatibili con **tar** di un altro sistema proprietario. Ciò potrebbe causare molti mal di testa, come quando l'archivio deve essere ripristinato in un sistema che non esiste più. Usate la versione GNU di **tar** in tutti i sistemi per prevenire che il vostro amministratore di sistema scoppi in lacrime. Linux utilizza sempre **tar** di GNU. Quando lavorate su altre macchine UNIX, inserite **tar --help** per scoprire quale versione state usando. Contattate il vostro amministratore di sistema se non vedete la parola GNU da qualche parte.

Nell'esempio seguente viene creato e spacchettato un archivio.

```
gaby:~> ls images/
me+tux.jpg nimf.jpg

gaby:~> tar cvf images-in-a-dir.tar images/
images/
images/nimf.jpg
images/me+tux.jpg

gaby:~> cd images

gaby:~/images> tar cvf images-without-a-dir.tar *.jpg
me+tux.jpg
nimf.jpg

gaby:~/images> cd

gaby:~> ls */*.tar
images/images-without-a-dir.tar

gaby:~> ls *.tar
images-in-a-dir.tar

gaby:~> tar xvf images-in-a-dir.tar
images/
images/nimf.jpg
images/me+tux.jpg

gaby:~> tar tvf images/images-without-dir.tar
-rw-r--r-- gaby/gaby 42888 1999-06-30 20:52:25 me+tux.jpg
-rw-r--r-- gaby/gaby   7578 2000-01-26 12:58:46 nimf.jpg

gaby:~> tar xvf images/images-without-a-dir.tar
me+tux.jpg
nimf.jpg

gaby:~> ls *.jpg
me+tux.jpg nimf.jpg
```

Questo esempio illustra anche la differenza tra una directory archiviata con tar e un complesso di file archiviati con tar. E' consigliabile comprimere le directory solamente, cosicché i file non sprizzeranno ovunque nel momento dello spacchettamento della tarball (cosa che potrebbe avvenire in un altro sistema dove non potreste sapere quali file ci siano già là e quali siano quelli

dell'archivio).

Quando una unità a nastro è connessa al vostro computer ed è stata configurata dall'amministratore di sistema, i nomi dei file terminanti in `.tar` vengono rimpiazzati con il nome della periferica a nastro, per esempio:

tar cvf /dev/tape mail/

La directory `mail` e tutti i file che contiene vengono compressi in un file che viene immediatamente scritto su nastro. Un elenco dei contenuti ci viene mostrato perché abbiamo usato l'opzione "verboso".

9.1.1.2. Copie di sicurezza incrementali con tar

Lo strumento **tar** supporta la creazione di backup incrementali, usando l'opzione -N. Con tale opzione potete specificare una data e **tar** controllerà la marca temporale (timestamp) di tutti i file specificati rispetto a questa data. Se i file sono cambiati più recentemente di tale data, saranno inclusi nella copia di sicurezza. L'esempio seguente utilizza il tempo di creazione di un precedente archivio come valore della data. Per prima cosa viene creato l'archivio iniziale e viene mostrata la marca temporale del file di backup. Poi viene creato un nuovo file di cui vogliamo fare un nuovo backup, che contenga solo questo nuovo file:

```
jimmy:~> tar cvpf /var/tmp/javaproggies.tar java/*.java
java/btw.java
java/error.java
java/hello.java
java/income2.java
java/income.java
java/inputdevice.java
java/input.java
java/master.java
java/method1.java
java/mood.java
java/moodywaitress.java
java/test3.java
java/TestOne.java
java/TestTwo.java
java/Vehicle.java

jimmy:~> ls -l /var/tmp/javaproggies.tar
-rw-rw-r-- 1 jimmy    jimmy  10240 Jan 21 11:58 /var/tmp/javaproggies.tar

jimmy:~> touch java/newprog.java

jimmy:~> tar -N /var/tmp/javaproggies.tar \
-cvp /var/tmp/incremental1-javaproggies.tar java/*.java 2> /dev/null
java/newprog.java

jimmy:~> cd /var/tmp/

jimmy:~> tar xvf incremental1-javaproggies.tar
java/newprog.java
```

Gli errori standard vengono rediretti verso `/dev/null`: se non fate ciò, **tar** stamperà un messaggio per ogni file non modificato dicendovi che non sarà eliminato.

Questo modo di funzionare ha lo svantaggio che si basa sul tempo di creazione dei file: dite che scaricate un archivio nella directory contenente le vostre copie di sicurezza e l'archivio contiene file che sono stati creati due anni fa. Quando viene controllata la data di creazione di quei file rispetto al tempo di creazione dell'archivio iniziale, i nuovi file sembreranno in realtà vecchi a **tar** e non saranno inclusi in un backup incrementale fatto usando l'opzione `-N`.

Una scelta migliore potrebbe essere l'opzione `-g`, che creerà un elenco di file da archiviare. Nel momento della copia di sicurezza incrementale, i file saranno confrontati con questo elenco. Funziona così:

```
jimmy:~> tar cvpf work-20030121.tar -g snapshot-20030121 work/
work/
work/file1
work/file2
work/file3

jimmy:~> file snapshot-20030121
snapshot-20030121: ASCII text
```

Il giorno dopo l'utente *jimmy* lavora ancora un po' su `file3` e crea `file4`. Al termine della giornata effettua una nuova copia di sicurezza:

```
jimmy:~> tar cvpf work-20030122.tar -g snapshot-20030121 work/
work/
work/file3
work/file4
```

Questi sono alcuni semplici esempi, ma potreste usare anche questo tipo di comando in un processo cron (v. Sezione 4.4.4.) che indica per esempio un file di snapshot per il backup settimanale e uno per quello giornaliero. I file di snapshot verranno rimpiazzati al momento dei backup completi in tal caso.

Maggiori informazioni si possono trovare nella documentazione di **tar**.

La vera sostanza

Come probabilmente potete notare, **tar** va bene quando stiamo discorrendo di una semplice directory, un insieme di file che stanno assieme. Comunque esistono degli strumenti che sono più facili da gestire quando volete archiviare intere partizioni, dischi o progetti più grandi. Noi parliamo qui solo di **tar** perché è uno strumento molto diffuso per distribuire archivi. Capiterà abbastanza spesso che avrete bisogno di installare un software che trovate nella cosiddetta "tarball compressa". Date uno sguardo alla Sezione 9.3. per un modo più semplice di effettuare copie di sicurezza regolari.

9.1.1.3. Compressione e spacchettamento con gzip o bzip2

I dati, comprese le tarball, possono essere compressi usando gli strumenti zip. Il comando **gzip** aggiungerà il suffisso .gz al nome del file e rimuoverà il file originale.

```
jimmy:~> ls -la | grep tar
-rw-rw-r-- 1 jimmy jimmy     61440 Jun  6 14:08 images-without-dir.tar

jimmy:~> gzip images-without-dir.tar
```

```
jimmy:~> ls -la images-without-dir.tar.gz
-rw-rw-r-- 1 jimmy jimmy     50562 Jun 6 14:08 images-without-dir.tar.gz
```

Decomprimete i file gzip con l'opzione `-d`.

bzip2 funziona in modo simile, ma usa un avanzato algoritmo di compressione, cosicché genera file più piccoli. Guardate le pagine info di **bzip2** per maggiori dettagli.

I pacchetti di Software Linux vengono spesso distribuiti in una tarball compressa con **gzip**. La cosa importante da fare dopo aver spacchettato questo genere di archivi è trovare README e leggerlo. In genere conterrà istruzioni per l'installazione del pacchetto.

Il comando GNU **tar** tiene in considerazione i file gzip. Usate il comando

tar `zxvf file.tar.gz`

per scompattare e dearchiviare i file `.tar.gz` o `.tgz`. Usate invece

tar `jxvf file.tar.bz2`

per spacchettare gli archivi **tar** che sono stati compressi con **bzip2**.

9.1.1.4. Gli archivi Java

Il progetto GNU ci mette a disposizione lo strumento **jar** per generare archivi Java. Si tratta di un'applicazione Java che combina molteplici file in un singolo file d'archivio JAR. Pur essendo uno strumento di uso generale per l'archiviazione e la compressione (basato sul formato di compressione ZIP e ZLIB), **jar** era stato concepito principalmente per l'impacchettamento di codice Java, applet e/o applicazioni, in un singolo file. I componenti di una applicazione Java, combinati in un solo file, possono essere scaricati molto più rapidamente.

Diversamente da **tar**, **jar** normalmente comprime, indipendentemente da altri strumenti (in quanto si tratta in sostanza della versione Java di **zip**). Inoltre permette inserimenti individuali in un archivio che l'autore sigla, in modo che le origini possono essere certificate.

La sintassi è quasi uguale a quella del comando **tar**. Ci riferiremo a **info *`jar`*** per differenze specifiche.

tar, jar e i collegamenti simbolici
Una funzione degna di nota, trascurata nella documentazione standard, è che **jar** seguirà i collegamenti simbolici. I dati a cui puntano questi collegamenti verranno ricompresi nell'archivio. Di base in **tar** si archiviano solo i collegamenti simbolici, ma questo comportamento può essere modificato con l'opzione `-h`.

9.1.1.5. Trasportare i vostri dati

Salvare copie dei vostri dati in un altro host è una semplice ma accurata maniera per fare dei backup. Guardate il Capitolo 10 per maggiori informazioni su **scp**, **ftp** ed altri ancora.

Nella prossima sezione tratteremo delle periferiche locali di backup.

9.2. Spostare i vostri dati verso un'unità di backup

9.2.1. Copiare su un disco floppy

9.2.1.1. Formattazione del floppy

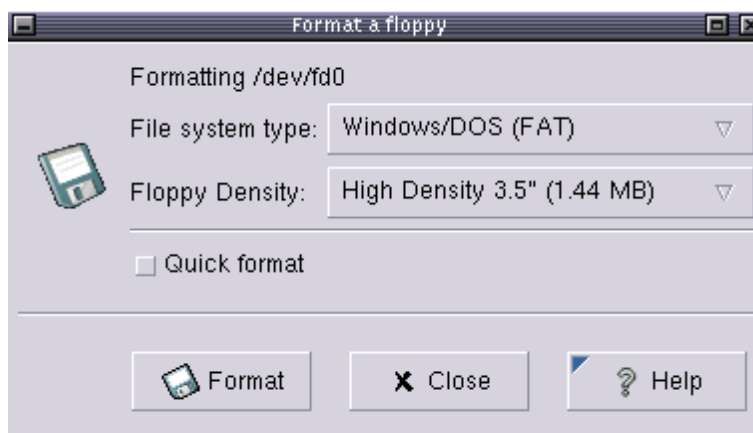
In molti sistemi Linux gli utenti hanno accesso alla periferica del disco floppy. Il nome della periferica può variare in base alla dimensione ed al numero di unità floppy: contattate l'amministratore di sistema se non ne siete sicuri. In alcuni sistemi ci sarà molto facilmente un collegamento `/dev/floppy` che punta alla periferica corretta, probabilmente `/dev/fd0` (unità floppy autorilevata) o `/dev/fd0H1440` (impostata per floppy da 1,44MB).

fdformat è lo strumento per la formattazione a basso livello dei dischi floppy. Usa come opzione il nome dell'unità del disco floppy. **fdformat** restituirà un errore nel caso in cui il floppy sia protetto da scrittura.

```
emma:~> fdformat /dev/fd0H1440
Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.
Formatting ... done
Verifying ... done
emma:~>
```

Il comando **mformat** (dal pacchetto mtools) viene utilizzato per creare dischetti DOS-compatibili, che poi si possono usare con **mcopy**, **mdir** ed altri comandi m.

Sono disponibili anche degli strumenti grafici.

Figura 9-1. Il formattatore di floppy.

Dopo che il dischetto è stato formattato, è possibile montarlo nel file system ed utilizzarlo come una

normale, seppure piccola, directory, normalmente attraverso l'inserimento di `/dev/floppy`.

Potreste averne bisogno: installate il programma di utilità **mkbootdisk** che crea un floppy da cui il sistema in uso può essere avviato.

9.2.1.2. Utilizzo di dd per copiare i dati

Il comando **dd** può essere impiegato per scrivere dei dati su un disco, oppure per copiarli nuovamente da questo, in base alle fornite unità di ingresso e uscita. Un esempio:

```
gaby:~> dd if=images-without-dir.tar.gz of=/dev/fd0H1440
98+1 records in
98+1 records out

gaby~> dd if=/dev/fd0H1440 of=/var/tmp/images.tar.gz
2880+0 records in
2880+0 records out

gaby:~> ls /var/tmp/images*
/var/tmp/images.tar.gz
```

Notate che la duplicazione viene eseguita su un'unità non montata. I floppy creati usando questo metodo non saranno montabili nel file system, ma questo è naturalmente il modo per creare dischi d'avvio o di ripristino. Per maggiori informazioni sulle possibilità di **dd**, leggete le pagine man.

Questo strumento fa parte del pacchetto GNU *coreutils*.

Duplicazione dei dischi
Il comando **dd** può essere usato anche per fare una copia grezza di un intero disco rigido.

9.2.2. Fare una copia con un masterizzatore di CD

In alcuni sistemi gli utenti hanno i permessi per utilizzare il masterizzatore di CD. Innanzitutto i vostri dati devono essere formattati. Usate il comando **mkisofs** per fare ciò nella directory contenente i file di cui intendete fare una copia di sicurezza. Controllate con **df** che ci sia abbastanza spazio disponibile sul disco perché verrà creato un nuovo file dalle dimensioni circa della directory corrente:

```
[rose@blob recordables] df -h .
Filesystem            Size Used Avail Use% Mounted on
/dev/hde5              19G   15G 3.2G 82% /home

[rose@blob recordables] du -h -s .
325M    .

[rose@blob recordables] mkisofs -J -r -o cd.iso .
<--snap-->
making a lot of conversions
<--/snap-->
98.95% done, estimate finish Fri Apr 5 13:54:25 2002
Total translation table size: 0
Total rockridge attributes bytes: 35971
Total directory bytes: 94208
Path table size(bytes): 452
```

```
Max brk space used 37e84
166768 extents written (325 Mb)
```

Le opzioni `-J` e `-r` vengono usate per rendere montabile il CD-ROM su differenti sistemi (vedete le pagine man per maggiori dettagli). Dopo di ciò, il CD può essere creato utilizzando lo strumento **cdrecord** con le appropriate opzioni:

```
[rose@blob recordables] cdrecord -dev 0,0,0 -speed=8 cd.iso
Cdrecord 1.10 (i686-pc-linux-gnu) (C) 1995-2001 Joerg Schilling
scsidev: '0,0,0'
scsibus: 0 target: 0 lun: 0
Linux sg driver version: 3.1.20
Using libscg version 'schily-0.5'
Device type    : Removable CD-ROM
Version        :0
Response Format: 1
Vendor_info    : 'HP      '
Identification : 'CD-Writer+ 8100 '
Revision       : '1.0g'
Device seems to be: Generic mmc CD-RW.
Using generic SCSI-3/mmc CD-R driver (mmc_cdr).
Driver flags   : SWABAUDIO
Starting to write CD/DVD at speed 4 in write mode for single session.
Last chance to quit, starting real write in 0 seconds.
Operation starts.
```

Ora, a seconda del vostro masterizzatore, avete il tempo per fumare, mangiare un salutare pezzo di frutta e/o bere un caffè. Al termine del processo otterrete un messaggio di conferma:

```
Track 01: Total bytes read/written: 341540864/341540864
          (166768 sectors).
```

Ci sono alcuni strumenti grafici a disposizione per rendere l'operazione più semplice. Uno tra i maggiormente diffusi è **xcdroast**, che è disponibile liberamente sul sito web di X-CD-ROAST ed è incluso nella maggior parte dei sistemi e nella directory GNU. Sia KDE che Gnome sono dotati di programmi per masterizzare i vostri CD personali.

9.2.3. Copie di sicurezza su/da unità jazz, periferiche USB e simili

Queste periferiche vengono comunemente montate nel file system. Dopo la procedura di montaggio, sono accessibili come normali directory, cosicché potete usare i comandi standard per manipolare i file.

Nell'esempio seguente delle immagini vengono copiate da una fotocamera USB ad un disco rigido:

```
robin:~> mount /mnt/camera

robin:~> mount | grep camera
/dev/sda1 on /mnt/camera type vfat (rw,nosuid,nodev)
```

Se la fotocamera è l'unica periferica di memorizzazione USB che avete collegato al vostro sistema, questo è sicuro. Ma tenete in mente che alle periferiche USB vengono assegnate delle voci in `/dev` quando si connettono al sistema. Così se prima collegate al vostro sistema una memoria USB, questa sarà inserita come `/dev/sda`, e se poi collegate dopo di questa una fotocamera,

quest'ultima sarà assegnata a `/dev/sdb` (ammesso che non abbiate dischi SCSI, che sarebbero anch'essi su `/dev/sd*`). Nei sistemi più recenti, a partire dal kernel 2.6, un sistema di collegamento a caldo chiamato HAL (Hardware Abstraction Layer) assicura che gli utenti non debbano trafficare con questo compito oneroso. Se volete verificare dove si trovi la vostra periferica, battete **dmesg** dopo averla attaccata.

Ora potete copiare i file:

```
robin:~> cp –R /mnt/camera/* images/

robin:~> umount /mnt/camera
```

Nello stesso modo una unità jazz può essere montata su `/mnt/jazz`.

Dovrebbero essere aggiunte le linee appropriate in `/etc/modules.conf` e `/etc/fstab` per far funzionare ciò. Riportatevi agli specifici HOWTO degli hardware per maggiori informazioni. Su sistemi con un kernel 2.6.x o superiore, potreste anche dover controllare le pagine man di **modprobe** e `modprobe.conf`.

9.2.4. Copie di sicurezza con una periferica a nastro

Questa operazione va fatta con **tar** (v. sopra). Lo strumento **mt** è utilizzato per controllare la periferica a nastro magnetico, come `/dev/st0`. Interi libri sono stati scritti sul backup a nastro, nonostante ciò, fate riferimento al nostro elenco di lettura in Appendice B per maggiori informazioni. Tenente in mente che i database potrebbero aver bisogno di altri procedure di backup a causa delle loro architetture.

I comandi giusti per le copie di sicurezza vengono normalmente inseriti in una delle directory *cron* allo scopo di mandarli in esecuzione con regolarità. In ambiti maggiori la suite liberamente disponibile Amanda o una soluzione commerciale possono essere introdotte per effettuare backup su molte macchine. Comunque l'uso dei nastri è un compito da amministratore di sistema che va oltre lo scopo di questo documento.

9.2.5. Strumenti dalla vostra distribuzione

La maggioranza delle distribuzioni Linux offre i propri strumenti per rendervi semplice la vita.
Una breve panoramica:

- Suse: ora Yast comprende copie di sicurezza estese e moduli di ripristino.
- RedHat: lo strumento File Roller fornisce una gestione visuale degli archivi (compressi). Sembra che siano in favore dello strumento X-CD-Roast per spostare le copie di sicurezza su un dispositivo esterno.
- Mandrake: X-CD-Roast.
- La maggioranza delle distribuzioni giunge fornita dei programmi di utilità BSD **dump** e **restore** per effettuare dei backup dei file system *ext2* e *ext3*. Tale strumento può scrivere in una varietà di dispositivi e, letteralmente, duplica il (i) file system bit per bit sulla periferica indicata. Come **dd**, questo permette di

creare copie di sicurezza di tipi speciali di file come quelli in `/dev`.

9.3. Uso di rsync

9.3.1. Introduzione

Il programma **rsync** è uno strumento rapido e flessibile per il backup remoto. E' comune nei sistemi UNIX e simil-UNIX, facile da configurare e da usare negli script. Sebbene la *r* in **rsync** stia per "remoto", non dovete prendere tutto ciò alla lettera. Il vostro dispositivo "remoto" potrebbe essere solo una periferica USB di massa o un'altra partizione del vostro disco rigido: non dovete per forza avere due macchine separate.

9.3.2. Un esempio: rsync su una periferica USB di massa

Come trattato nella Sezione 3.1.2.3., dobbiamo per prima cosa montare la periferica. Ciò andrebbe fatto, possibilmente, in qualità di *root*:

```
root@theserver# mkdir /mnt/usbstore

root@theserver# mount –t vfat /dev/sda1 /mnt/usbstore
```

Facilità d'uso
Moltissime distribuzioni concedono agli utenti non privilegiati l'accesso alle periferiche removibili e montano i dispositivi USB, CD-ROM ed altri in modo automatico.

Notate che questa istruzione richiede l'installazione del supporto USB nel vostro sistema. Guardate USB Guide per un aiuto se questo non funziona. Verificate con **dmesg** che `/dev/sda1` sia naturalmente la periferica da montare.

Dopo potete avviare l'attuale backup, per esempio, della directory `/home/karl`:

```
karl@theserver:~> rsync –avz /home/karl /mnt/usbstore
```

Come il solito, fate riferimento alle pagine man per maggiori informazioni.

9.4. Crittografia

9.4.1. Note generali

9.4.1.1. Perché dovreste crittografare i dati?

La crittografia è sinonimo di segretezza. Nel contesto delle copie di sicurezza la crittografia può essere molto utile, per esempio se avete necessità di lasciare i vostri dati salvati in un posto in cui non potete controllare gli accessi, come il server del vostro provider.

A parte ciò, la crittografia può essere adottata pure con la posta elettronica: normalmente questa

non è criptata e spesso viene inviata allo scoperto sulla rete o su Internet. Se il vostro messaggio contiene delle informazioni sensibili, allora è meglio crittografarlo.

9.4.1.2. GNU Privacy Guard

Nei sistemi Linux potrete trovare GnuPG, Gnu Privacy Guard, che è un complesso di programmi compatibili con gli strumenti PGP (Pretty Good Privacy), che sono disponibili commercialmente.

In questa guida tratteremo soltanto dell'uso più elementare degli strumenti di crittografia e mostreremo ciò di cui avrete bisogno per generare una chiave di codifica e per usarla da voi stessi nella crittografia dei dati (che poi potrete conservare tranquillamente in un luogo pubblico). Indicazioni per un uso maggiormente avanzato possono essere rintracciate nelle pagine man dei vari comandi.

9.4.2. La generazione di una chiave

Prima di cominciare a criptare i vostri dati, dovete creare un paio di chiavi. La coppia consiste in una chiave privata ed in una pubblica. Potete inviare la chiave pubblica ai corrispondenti, che possono usarla per criptare i dati a voi desinati, che decripterete con la vostra chiave privata. Tenete sempre la chiave, non condividetela con qualunque altro, o essi saranno in grado di decriptare i dati destinati solo a voi. Esclusivamente per essere certi che non accadano incidenti, la chiave privata è protetta da una password. La coppia di chiavi viene creata usando questo comando:

```
willy@ubuntu:~$ gpg --key-gen
gpg (GnuPG) 1.4.2.2; Copyright (C)  2005 Free Software Foundation, Inc.
This program comes with ABSOLUTELY  NO WARRANTY.
This is free software, and you are  welcome to redistribute it
under certain conditions. See the   file COPYING for details.

gpg: directory `/home/willy.gnupg' created
gpg: new configuration file `/home/willy/.gnupg/gpg.conf' created
gpg: WARNING: options in `/home/willy/.gnupg/gpg.conf' are not yet
 active during this run
gpg: keyring `/home/willy/.gnupg/secring.gpg' created
gpg: keyring `/home/willy/.gnupg/pubring.gpg' created
Please select what kind of key you want:
    (1) DSA and Elgamal (default)
    (2) DSA (sign only)
    (5) RSA (sign only)
Your selection? 1
DSA keypair will have 1024 bits.
ELG-E keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048) 4096
Requested keysize is 4096 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n> = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n month
      <n>y = key expires in n years
Key is valid for? (0) 0
Key does not expire at all
Is this correct? (y/N) y

You need a user ID to identify your key; the software constructs the
user ID from the Real Name, Comment and Email Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

Real name: Willy De Wandel
```

```
Email address: wdw@mvg.vl
Comment: Willem
You selected this USER-ID:
    "Willy De Wandel (Willem) <wdw@mvg.vl>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
You need a Passphrase to protect your secret key.

Passphrase:
```

Ora inserite la vostra password. Questa può essere una frase, la più lunga, la migliore: l'unica condizione è che dovreste ricordarla ogni volta. Per verifica dovete inserire la stessa frase un'altra volta.

Ora viene generata la coppia di chiavi da un programma che produce numeri casuali e che, fra gli altri fattori, viene alimentato dai dati dell'attività del sistema. Cosicché è una buona idea avviare ora alcuni programmi, muovere il cursore del mouse o battere alcuni caratteri a caso in una finestra di terminale. In questo modo saranno molto più grandi le possibilità di creare un numero che contenga molte cifre differenti e la chiave sarà molto difficile da individuare.

9.4.3. A proposito della vostra chiave

Dopo che è stata creata la vostra chiave, riceverete u messaggio circa l'*impronta digitale* (*fingerprint*). Si tratta di una sequenza di quaranta numeri esadecimali, così lunga da essere molto, molto difficile generarne una seconda uguale su qualsiasi computer. Potete essere piuttosto sicuri che questa sia una sequenza unica. La forma abbreviata di questa chiave consiste nel vostro nome seguito dagli ultimi 8 numeri esadecimali.

Potete ottenere informazioni sulla vostra chiave nel modo seguente:

```
willy@ubuntu:~$ gpg --list-keys
/home/willy/.gnupg/pubring.gpg
------------------------------

pub     1024D/BF5C3DBB 2006-08-08
uid                    Willy De Wandel (Willem) <wdw@mvg.vl>
sub  4096g/A3449CF7 2006-08-08
```

L'*ID di chiave* (*key ID*) per questa chiave è "BF5C3DBB". Potete inviare i vostri ID di chiave e nome ad un *server di chiavi* (*key server*), cosicché altre persone possano avere queste informazioni su di voi e le usino per criptare i dati a voi diretti. In alternativa, potreste inviare la vostra chiave pubblica direttamente ai soggetti che ne hanno bisogno. La parte pubblica della chiave è una lunga serie du numeri che potete vedere usando l'opzione `--export` con il comando **gpg**:

gpg --export -a

Comunque, per quanto riguarda questa guida, supponiamo che abbiate bisogno della chiave solo per codificare e decodificare dati per voi stessi. Leggete le pagine man di **pgp** se volete saperne di più.

9.4.4. Crittografia dei dati

Ora potete criptare un archivio `.tar` o compresso, prima di salvarlo su un supporto di salvataggio

o di trasportarlo sul server di backup. Usate il comando **gpg** come qui di seguito:

gpg -e -r *(parte di) uidarchivio*

L'opzione `-e` dice a **gpg** di criptare, l'opzione `-r` indica per chi si cripta. Tenete in mente che solo i nomi utente che seguono tale opzione -r saranno in grado di decodificare nuovamente i dati. Un esempio:

```
willy@ubuntu:~$ gpg -e -r Willy /var/tmp/home-willy-20060808.tar
```

9.4.5. Decodifica dei file

Utilizzando l'opzione `-d`, potete decriptare i file che sono stati codificati per voi. I dati vi scorreranno sullo schermo, ma rimarrà una copia criptata su disco. Così per i file diversi da quelli di semplice testo, dovrete salvare i dati decodificati per poterli vedere con il programma appropriato. Ciò di effettua ricorrendo all'opzione `-o` con il comando **gpg**:

```
willy@ubuntu:~$ gpg -d -o /var/tmp/home-willy-decrypt.tar /var/tmp/home-willy-20060808.tar.gpg

You need a passphrase to unlock the secret key for
user: "Willy De Wandel (Willem) <wdw@mvg.vl>"
4096 ELG-E key, ID A3449CF7, created 2006-08-08 (main key ID BF5C3DBB)

gpg: encrypted with 4096-bit ELG-E key, ID A3449CF7, created 2006-08-08
      "Willy De Wandel (Willem) <wdw@mvg.vl>"
```

Niente password=niente dati

Se vi scordate la password, i dati sono persi. Neppure l'amministratore di sistema sarà capace di decrittare i dati. Ecco perché talvolta una copia delle chiavi importanti viene conservata in una cassetta di sicurezza di una banca.

9.5. Sommario

Qui c'è un elenco dei comandi riguardanti il backup dei file:

Tabella 9-1. Nuovi comandi nel capitolo 9: La copia di sicurezza

Comando	Significato
bzip2	Un compressore di file a ordinamento a blocchi
cdrecord	Registra Compact Disk audio o dati da un originale
dd	Converte e copia un file
fdformat	Formatta a basso livello di un dischetto floppy
gpg	Codifica e decodifica i dati.
gzip	Comprime o espande file

Comando	Significato
mcopy	Copia file MSDOS verso/da UNIX
mdir	Mostra una directory MSDOS
mformat	Aggiunge un file system MSDOS ad un dischetto floppy formattato a basso livello
mkbootdisk	Crea un dischetto di avvio indipendente per far girare il sistema
mount	Monta un file system (integratelo con il file system corrente connettendolo in un punto di montaggio)
rsync	Sincronizza le directory
tar	Programma di utilità per l'archiviazione su nastro, usato anche per creare archivi su disco invece che su nastro
umount	Smonta i file system

9.6. Esercizi

- Fate una copia della vostra directory personale verso `/var/tmp` usando il comando **tar**. Successivamente comprimete il file usando **gzip** o **bzip2**: fatelo in modo che ne risulti un corretto file archiviato con **tar**, uno che non crei confusione una volta scompattato.
- Formattate un floppy e mettete alcuni file della vostra directory personale su di esso. Scambiate i floppy con un altro studente e ripristinate il suo floppy nella vostra directory personale
- Formattate il floppy in DOS. Usate gli *mtools* per mettere e cancellare file su di esso.
- Cosa succede ad un dischetto non formattato quando lo volete montare nel file system?
- Se avete una memoria USB, provate a registrarci dentro un file.
- Usando **rsync,** fate una copia della vostra directory personale in un altro file system locale o remoto.
- Quando si lasciano file in un server di rete, è meglio criptarli. Create un archivio `tar` nella vostra directory personale e criptatelo.

Capitolo 10. Le reti

Quando si parla di reti, Linux è il vostro sistema operativo d'elezione, non solo perché le reti sono strettamente integrate nel S.O. stesso e sono disponibili un'ampia varietà di strumenti ed applicazioni liberi, ma anche per la robustezza sotto carichi pesanti che può essere raggiunta solo dopo anni di debug e prove in un progetto Open Source.

Sono stati scritti scaffali di libri pieni di informazioni su Linux e le reti, ma in questo capitolo proveremo a fornire una panoramica. Dopo averlo completato, saprete di più su

- i protocolli di rete supportati
- i file di configurazione della rete
- i comandi per configurare e provare la rete
- i demoni e i programmi clienti che abilitano differenti applicazioni di rete
- la condivisione e la stampa di file
- l'esecuzione remota di comandi e applicazioni
- l'interconnessione base di rete
- l'esecuzione sicura di applicazione remote
- i firewall e l'individuazione delle intrusioni

10.1. Panoramica sulle reti

10.1.1. Il modello OSI

Un protocollo è, per semplificare, un insieme di regole per le comunicazioni.

Per inviare dati in una rete, per esempio una lettera elettronica dal vostro computer ad un computer all'altra estremità del mondo, molti hardware e software differenti devono lavorare insieme.

Tutti queste parti di hardware e programmi software differenti parlano linguaggi diversi. Immaginate il vostro programma di posta elettronica: è in grado dialogare con il sistema operativo del computer attraverso uno specifico protocollo ma non è capace di parlare con l'hardware del computer. Ci serve uno speciale programma nel sistema operativo che svolga tale compito. A sua volta il computer deve essere in grado di comunicare con la linea telefonica od altri metodi di collegamento Internet. E, dietro le quinte, l'hardware di collegamento alla rete deve poter colloquiare per trasmettere il vostro messaggio da un apparecchio all'altro per tutto il percorso sino al computer di destinazione.

Tutti questi tipi differenti di protocolli di comunicazione sono classificati in 7 livelli, noti come *Open Systems Interconnections Reference Model*, in breve *Modello OSI*. Per una facile comprensione, tale modello viene ridotto ad una descrizione di protocollo a 4 livelli, come descritto nella tabella seguente:

Tabella 10-1. Il Modello OSI semplificato

Nome del livello	Protocolli del livello
Livello applicazioni	HTTP, DNS, SMTP, POP, ...
Livello di trasporto	TCP, UDP
Livello di rete	IP, IPv6
Livello di accesso alla rete	PPP, PPPoE, Ethernet

Ciascun livello può utilizzare le funzionalità del livello inferiore; ogni livello può solo esportare le funzionalità al livello superiore. In altre parole: i livelli comunicano solo con quelli adiacenti. Riprendiamo l'esempio del messaggio di posta elettronica: lo inviate per mezzo del livello delle applicazioni. Nel vostro computer scende attraverso i livelli di trasporto e rete. Il vostro computer lo pone in rete tramite il livello dell'accesso in rete. Questo è anche il livello che trasferirà il messaggio intorno al mondo. Giunto a destinazione, il computer ricevente accetterà il messaggio attraverso il proprio livello di rete e lo mostrerà al destinatario utilizzando i livelli di trasporto e delle applicazioni.

E' veramente molto più complicato
Le precedenti e successive sezioni sono ricomprese perché presto o tardi verrete in contatto con alcuni termini di reti; esse vi forniranno alcuni punti di partenza. Dovreste cercare i dettagli.

10.1.2. Alcuni popolari protocolli di rete

Linux supporta molti protocolli differenti di rete. Elenchiamo solo quelli più importanti:

10.1.2.1. TCP/IP

Il *Transport Control Protocol* e l'*Internet Protocol* sono due dei modi più popolari di comunicazione su Internet. Molte applicazioni, come i vostri programmi di navigazione e di posta elettronica, sono costruiti sopra questo complesso di protocolli.

Detto molto semplicemente, IP fornisce una soluzione per inviare pacchetti di informazioni da una macchina ad un'altra, mentre TCP assicura che i pacchetti siano disposti in flussi, in modo che pacchetti da diverse applicazioni non vengano mescolati e che siano inviati e ricevuti nell'ordine corretto.

Un buon punto di partenza per imparare di più su TCP e IP sono i seguenti documenti:

- **man 7 *`ip`***: descrive l'implementazione su Linux del protocollo IPv4 (essendo attualmente la versione 4 la più diffusa edizione del protocollo IP).
- **man 7 *`tcp`***: implementazione del protocollo TCP
- RFC793, RFC1122, RFC2001 per TCP, e RFC791, RFC1122 e RFC1112 per IP

  I documenti Request For Comment contengono le descrizioni degli standard, dei protocolli, delle applicazioni e delle implementazioni di rete. Questi documenti sono gestiti dalla Internet Engineering Task Force, una comunità internazionale che si occupa del

funzionamento di Internet senza intoppi, dell'evoluzione e dello sviluppo dell'architettura di Internet.

Il vostro ISP ha normalmente a disposizione un archivio di RFC, oppure potete navigare tra gli RFC via http://www.ietf.org/rfc.html.

10.1.2.2. TCP/IPv6

Nessuno si aspettava che Internet crescesse così in fretta come ha fatto. IP ha dimostrato di avere alcuni svantaggi quando in rete è presente un numero molto grande di computer, essendo di notevole importanza la disponibilità di indirizzi unici da assegnare ad ogni macchina connessa. Così IP versione 6 è stato congegnato per soddisfare le necessità dell'odierna Internet.

Sfortunatamente non tutte le applicazioni e i servizi supportano ancora IPv6. Attualmente è in corso una migrazione in molti ambienti che possono trarre beneficio dall'aggiornamento a IPv6. Per alcune applicazioni il vecchio protocollo è ancora in uso, mentre la nuova versione è già attiva per applicazioni che sono state aggiornate. Cosicché, quando verificate la vostra configurazione di rete, è possibile che qualche volta sia leggermente confusa dal momento che tutti i generi di misure possono essere state adottate per nascondere un protocollo dall'altro affinché i due non confondano le comunicazioni.

Maggiori informazioni si possono trovare nei documenti seguenti:

- **man *`7 ipv6`***: l'implementazione del protocollo IPv6 in Linux;
- RFC1883 descrive il protocollo IPv6.

10.1.2.3. PPP, SLIP, PLIP, PPPOE

Il kernel di Linux ha il supporto integrato per PPP (Point-to-Point-Protocol), SLIP (Serial Line IP), PLIP (Parallel Line IP) e PPP Over Ethernet. PPP è il modo più diffuso con cui gli utenti individuali accedono al loro ISP (Internet Service Provider), sebbene in aree densamente popolate sia spesso sostituito da PPPOE, PPP over Ethernet, il protocollo utilizzato nelle connessioni ADSL (Asymmetric Digital Subscriber Line).

La maggioranza delle distribuzioni Linux fornisce strumenti di facile uso per impostare una connessione Internet. L'unica cosa di cui avete sostanzialmente bisogno è un nome utente ed una password per connettervi al vostro Internet Service Provider (ISP) e un numero telefonico in caso di PPP. Questi dati vengono inseriti nello strumento grafico di configurazione, che assai probabilmente vi permetterà anche di avviare ed interrompere la connessione con il provider.

10.1.2.4. ISDN

Il kernel di Linux ha il supporto integrato ISDN. Isdn4linux controlla le schede ISDN per PC e può emulare un modem con l'insieme dei comandi Hayes (comandi "AT"). Le possibilità spaziano dal semplice uso di un programma di terminale ad una completa connessione a Internet.

Verificate la vostra documentazione di sistema.

10.1.2.5. AppleTalk

AppleTalk è il nome dello stack di interconnessione di Apple. Consente un tipo di rete peer-to-peer che fornisce funzionalità di base come la condivisione di file e stampanti. Ciascuna macchina può contemporaneamente agire in qualità di cliente e di server ed il software e l'hardware sono compresi in ogni computer Apple.

Linux fornisce connessioni complete a AppleTalk. Netatalk è una implementazione a livello kernel della Suite di Protocollo AppleTalk, in origine per sistemi BSD-derivati. Comprende il supporto per l'instradamento AppleTalk, funzioni di server per file system UNIX e AFS usando AppleShare, di server per stampanti UNIX e di accesso alle stampanti AppleTalk.

10.1.2.6. SMB/NMB

Per compatibilità con gli ambienti MS Windows, la suite Samba, comprendente il supporto per i protocolli NMB e SMB, può essere installata in qualsiasi sistema simil-UNIX. Il protocollo Server Message Block (chiamato anche Session Message Block, NetBIOS o protocollo LanManager) viene usato in MS Windows 3.11, NT, 95/98, 2K e XP per condividere file e stampanti.

Le funzioni base della suite Samba sono: condivisione dei drive Linux con le macchine Windows, accesso alle condivisioni SMB da macchine Linux, condivisione delle stampanti Linux con le macchine Windows e viceversa.

La maggioranza delle distribuzioni Linux fornisce un pacchetto *samba* che crea gran parte delle impostazioni del server e avvia **smbd**, il server Samba, e **nmbd**, il server netbios dei nomi, solitamente al momento dell'avvio. Samba può essere configurato in modalità grafica, attraverso un'interfaccia web o attraverso la linea di comando e i file di configurazione testuale. I demoni fanno in modo che una macchina Linux appaia come un host MS Windows in una finestra MS Windows My Network Places/Network Neighbourhood; una condivisione da una macchina Linux sarà indistinguibile da una di un qualsiasi altro host in ambiente MS Windows.

Maggiori informazioni si possono trovare nei posti seguenti:

- **man `smb.conf`**: descrive il formato del principale file di configurazione di Samba.
- Samba Project Documentation (o verificate il vostro mirror locale di samba.org) contiene una guida di facile lettura sull'installazione e la prova, che spiega pure come configurare il vostro server Samba come Primary Domain Controller. Qui sono anche a disposizione tutte le pagine man.

10.1.2.7. Miscellanea di protocolli

Linux ha anche il supporto per le reti radioamatoriali e WANX25, Frame Relay, ATM), per le connessioni all'infrarosso ed altre senza fili, ma dal momento che questi protocolli normalmente richiedono hardware speciale, non li tratteremo in questo documento.

10.2. Configurazioni ed informazioni di rete

10.2.1. Configurazione delle interfacce di rete

Tutte le grosse distribuzioni Linux di facile utilizzo sono dotate di vari strumenti grafici che consentono l'impostazione semplice del computer in una rete locale, la sua connessione ad un Internet Service Provider o l'accesso wireless (senza fili, via radio). Questi strumenti possono essere avviati dalla linea di comando o da un menu:

- La configurazione viene eseguita selezionando System+Administration−>Networking.
- RedHat viene con **redhat-config-network**, che ha un'interfaccia in modalità sia grafica che testuale.
- YAST o YAST2 di SuSE sono strumenti di configurazione generale.
- Mandrake/Mandriva arriva dotata di un assistente di configurazione di rete e Internet, che è preferibile avviare dal Centro di Controllo Mandrake.
- In sistemi Gnome: **gnome-network-preferences**.
- In sistemi KDE: **knetworkconf**.

La vostra documentazione di sistema fornisce consigli e informazioni in abbondanza circa la disponibilità e l'utilizzo di strumenti.

Le informazioni che dovrete fornire:

- per la connessione ad una rete locale, per esempio con i vostri computer domestici, o al lavoro: nome host, nome di dominio e indirizzo IP. Se volete impostare la vostra rete personale, prima meglio leggere qualcosa in più. Al lavoro questa informazione probabilmente viene assegnata al vostro computer automaticamente al momento dell'avvio. In caso di dubbi, è meglio non specificare alcuna informazione se non quelle necessarie;
- per la connessione ad Internet: nome utente e password per il vostro ISP, numero di telefono quando si utilizza un modem. Normalmente il vostro ISP vi attribuisce automaticamente un indirizzo IP e tutti gli altri dati necessari al funzionamento delle vostre applicazioni Internet.

10.2.2. I file di configurazione di rete

Gli strumenti grafici di aiuto modificano uno specifico insieme dei file di configurazione di rete utilizzando una coppia di comandi di base. I nomi esatti dei file di configurazione e la loro posizione nel file system dipendono largamente dalla vostra distribuzione e versione di Linux. Comunque, una coppia di file di configurazione sono comuni in tutti i sistemi UNIX:

10.2.2.1. /etc/hosts

Il file `/etc/hosts` contiene sempre l'indirizzo IP *localhost*, 127.0.0.1, che viene utilizzato per la comunicazione tra i processi. Non cancellate mai questa linea!. Qualche volta contiene gli indirizzi di host aggiuntivi che possono essere raggiunti senza utilizzare un servizio esterno di naming come il DNS (Domain Name Server).

Un semplice file `hosts` di esempio per una piccola rete domestica:

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1       localhost.localdomain   localhost
192.168.52.10   tux.mylan.com           tux
192.168.52.11   winxp.mylan.com         winxp
```

Leggete di più in **man hosts**.

10.2.2.2. /etc/resolv.conf

Il file `/etc/resolv.conf` configura l'accesso ad un server DNS (v. Sezione 10.3.7.). Questo file contiene il vostro nome di dominio ed il nome del (dei) server da contattare:

```
search mylan.com
nameserver 193.134.20.4
```

Leggete ulteriormente nella pagina man di `resolv.conf`.

10.2.2.3. /etc/nsswitch.conf

Il file `/etc/nsswitch.conf` definisce l'ordine con cui contattare vari servizi dei nomi. Per l'uso di Internet, è importante che *dns* appaia nella linea "hosts":

```
[bob@tux ~] grep hosts /etc/nsswitch.conf
hosts: files dns
```

Questo istruisce il vostro computer a cercare i nomi degli host e gli indirizzi IP prima nel file `/etc/hosts` e poi a contattare il server DNS se un certo host non compare nel file locale `hosts`. Altri possibili servizi dei nomi da contattare sono LDAP, NIS e NIS+.

Di più in **man nsswitch.conf**.

10.2.3. I comandi di configurazione delle reti

10.2.3.1. Il comando ip

Gli script specifici delle distribuzioni e gli strumenti grafici sono interfacce a **ip** (o **ifconfig** e **route** in sistemi più vecchi) per impostare la configurazione di rete del kernel.

Il comando **ip** viene utilizzato per assegnare indirizzi IP alle interfacce, per impostare gli instradamenti verso Internet e verso altre reti, per mostrare le configurazioni TCP/IP, ecc...

I seguenti comandi mostrano l'indirizzo IP e le informazioni dell'instradamento:

```
benny@home benny> ip addr show
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 brd 127.255.255.255 scope host lo
    inet6 ::1/128 scope host
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    link/ether 00:50:bf:7e:54:9a brd ff:ff:ff:ff:ff:ff
```

```
    inet 192.168.42.15/24 brd 192.168.42.255 scope global eth0
    inet6 fe80::250:bfff:fe7e:549a/10 scope link

benny@home benny> ip route show
192.168.42.0/24 dev eth0 scope link
127.0.0.0/8 dev lo scope link
default via 192.168.42.1 dev eth0
```

Cose da notare:

- Due interfacce di rete, anche in un sistema che ha un'unica scheda di rete: “lo” è il *local loop* usato per le comunicazioni interne; “eth0” è un nome comune di una *vera* interfaccia. Non cambiate mai la configurazione del local loop, in caso contrario la vostra macchina incomincerà a funzionare male! Le interfacce senza fili vengono abitualmente definite come “wlan0”; le interfacce modem come “ppp0”, ma potrebbero pure esserci altri nomi.
- Gli indirizzi IP segnati con “inet”: il local loop ha sempre 127.0.0.1, l'interfaccia fisica può avere qualsiasi altra combinazione.
- L'indirizzo hardware della vostra interfaccia, che potrebbe essere richiesto come parte della procedura di autenticazione per la connessione ad un network, viene indicato con “ether”. Il local loop ha sei paia di tutti zeri, il loop fisico ha sei coppie di caratteri esadecimali di cui i primi tre sono specifici del produttore.

10.2.3.2. Il comando ifconfig

Mentre **ip** è il modo più moderno per configurare un sistema Linux, **ifconfig** è ancora molto diffuso. Usateli senza opzioni per mostrare le informazioni dell'interfaccia di rete:

```
els@asus:~$ /sbin/ifconfig
eth0      Link encap:Ethernet HWaddr 00:50:70:31:2C:14
          inet addr:60.138.67.31 Bcast:66.255.255.255 Mask:255.255.255.192
          inet6 addr: fe80::250:70ff:fe31:2c14/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:31977764 errors:0 dropped:0 overruns:0 frame:0
           TX packets:51896866 errors:0 dropped:0 overruns:0 carrier:0
           collisions:802207 txqueuelen:1000
           RX bytes:2806974916 (2.6 GiB) TX bytes:2874632613 (2.6 GiB)
           Interrupt:11 Base address:0xec00
                                                                lo
           inet addr:127.0.0.1 Mask:255.0.0.0
           inet6 addr: ::1/128 Scope:Host
           UP LOOPBACK RUNNING MTU:16436 Metric:1
           RX packets:765762 errors:0 dropped:0 overruns:0 frame:0
           TX packets:765762 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:0
           RX bytes:624214573 (595.2 MiB) TX bytes:624214573 (595.2 MiB)
```

Pure qui notiamo gli aspetti più rilevanti della configurazione dell'interfaccia:

- L'indirizzo IP è indicato con “inet addr”.
- L'indirizzo hardware segue il tag “Hwaddr”.

Sia **ifconfig** che **ip** mostrano informazioni più dettagliate sulla configurazione e numerosi dati statistici circa ciascuna interfaccia e, potrebbe essere maggiormente importante, se quest'ultima è “UP” e “RUNNING”.

10.2.3.3. I comandi PCMCIA

Nel vostro portatile, che normalmente collegate alla rete aziendale utilizzando la connessione Ethernet integrata, ma che ora dovete configurare per una connessione telefonica a casa o in albergo, potreste aver bisogno di attivare la scheda PCMCIA. Ciò si effettua tramite l'utilità di controllo **cardctl** o con **pccardctl** nelle nuove distribuzioni.

Un esempio dell'uso:

cardctl ***`insert`***

Ora la scheda può essere configurata, sia con l'interfaccia grafica che con quella a linea di comando. Prima di estrarre la scheda usate questo comando:

cardctl ***`eject`***

Comunque una buona distribuzione dovrebbe fornire il supporto PCMCIA con gli strumenti di configurazione della rete, evitando agli utenti di dover eseguire manualmente i comandi PCMCIA.

10.2.3.4. Maggiori informazioni

Dibattere ulteriormente sulla configurazione di rete è al di fuori dello scopo di questo documento. La vostra fonte primaria di informazioni extra sono le pagine man per i servizi che volete impostare. Letture addizionali:

- Modem-HOWTO: aiuta nella scelta, connessione, configurazione, risoluzione dei problemi e comprensione dei modem analogici per i PC.
- Indice LDP HOWTO, sezione 4.4: elenco di HOWTO suddiviso in categorie circa le reti in generale, i protocolli, le dial-up, il DNS, le VPN, il bridging, l'instradamento, la sicurezza ed altro ancora.

Molti sistemi hanno una versione di file `ip-cref` (trovatelo utilizzando il comando **locate**): il formato PS di questo file è visibile, per esempio, con **gv**.

10.2.4. Nomi delle interfacce di rete

In una macchina Linux il nome di periferica *lo* o *local loop* è collegato con l'indirizzo interno 127.0.0.1. Se non è presente tale periferica, il computer passerà un brutto momento nel far funzionare le vostre applicazioni: è infatti sempre presente, anche in computer non collegati in rete.

La prima periferica ethernet, *eth0* nel caso di una tradizionale scheda di interfaccia di rete punta al vostro indirizzo locale LAN IP. Le normali macchine clienti hanno soltanto un'unica scheda di rete. I router, collegando le reti tra di loro, hanno una periferica di rete per ciascuna rete servita.

Se usate un modem per connettervi ad Internet, la vostra periferica di rete probabilmente verrà chiamata *ppp0*.

Esistono molti altri nomi, per esempio per le interfacce Virtual Private Network (VPN o rete privata virtuale), e numerose interfacce possono essere attive contemporaneamente, cosicché l'output dei

comandi **ifconfig** o **ip** potrebbe divenire piuttosto esteso quando non vengono utilizzate delle opzioni. Anche molteplici interfacce dello stesso genere possono essere attive. In tal caso vengono numerate in sequenza: la prima avrà il numero 0, la seconda un suffisso 1, la terza 2, e così via. Questo è il caso di molti server di applicazioni, di macchine dotate di configurazione failover, di router, firewall e molte ancora.

10.2.5. La configurazione del vostro host

A prescindere dal comando **ip** che mostra la configurazione di rete, c'è il comune comando **netstat** che ha molte opzioni ed è generalmente utile in qualsiasi sistema UNIX.

Le informazioni di instradamento possono essere mostrate con l'opzione `-nr` del comando **netstat** :

```
bob:~> netstat -nr
Kernel IP routing table
Destination Gateway       Genmask       Flags MSS Window irtt Iface
192.168.42.0 0.0.0.0      255.255.255.0 U      40 0          0 eth0
127.0.0.0    0.0.0.0      255.0.0.0     U      40 0          0 lo
0.0.0.0      192.168.42.1 0.0.0.0       UG     40 0          0 eth0
```

Questa è una tipica macchina cliente in una rete IP: ha solo un'interfaccia di rete, *eth0*. L'interfaccia *lo* è il local loop.

La maniera moderna
La maniera attuale per ottenere queste informazioni dal vostro sistema è di ricorrere al comando **ip**:

ip ***`route show`***

Quando tale macchina tenterà di contattare un host che si trova in una rete diversa dalla sua, indicata dalla linea iniziante con 0.0.0.0, essa invierà le richieste di connessione alla macchina (router) con indirizzo IP 192.168.42.1 ed utilizzerà la propria interfaccia primaria eth0 per fare ciò.

Gli host che si trovano sulla stessa rete (la linea iniziante con 192.168.42.0) verranno altresì contattati attraverso l'interfaccia di rete primaria, ma non sarà necessario un router: i dati saranno immessi semplicemente in rete.

Le macchine possono avere tabelle di instradamento più complicate di questa, con molte coppie di “Destinazione-Gateway” per connettersi a diverse reti. Se vi capita l'occasione di collegarvi ad un server di applicazioni (per esempio al lavoro), è piuttosto educativo verificare le informazioni di instradamento.

10.2.6. Altri host

Un impressionante numero di strumenti è mirato alla gestione delle reti ed all'amministrazione remota di macchine Linux. Il vostro mirror locale di software Linux ve ne offrirà in abbondanza di questi. Ci porterebbe troppo lontano trattarli in questo documento, così fate riferimento per favore alla documentazione specifica del programma.

Parleremo in questa sezione soltanto di alcuni comuni strumenti testuali di UNIX/Linux.

10.2.6.1. Il comando host

Per mostrare le informazioni sugli host o i domini, usate il comando **host**:

```
[emmy@pc10 emmy]$ host www.eunet.be
www.eunet.be. has address 193.74.208.177

[emmy@pc10 emmy]$ host -t any eunet.be
eunet.be. SOA dns.eunet.be. hostmaster.Belgium.EU.net.
  2002021300 28800 7200 604800 86400
eunet.be. mail is handled by 50 pophost.eunet.be.
eunet.be. name server ns.EU.net.
eunet.be. name server dns.eunet.be.
```

Simili informazioni possono essere visualizzate usando il comando **dig**, che dà nozioni aggiuntive su come i record vengono registrati nel server dei nomi.

10.2.6.2. Il comando ping

Per controllare se un host è attivo utilizzate **ping**. Se il vostro sistema è configurato per inviare più di un pacchetto, interrompete **ping** con la combinazione di tasti **Ctrl + C**:

```
[emmy@pc10 emmy]$ ping a.host.be
PING a.host.be (1.2.8.3) from 80.20.84.26: 56(84) bytes of data.
64 bytes from a.host.be(1.2.8.3):icmp_seq=0 ttl=244 time=99.977msec
--- a.host.be ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max/mdev = 99.977/99.977/99.977/0.000 ms
```

10.2.6.3. Il comando traceroute

Per controllare il percorso seguito dai pacchetti verso un host di rete, usate il comando **traceroute**:

```
[emmy@pc10 emmy]$ /usr/sbin/traceroute www.eunet.be
traceroute to www.eunet.be(193.74.208.177),30 hops max,38b packets
1 blob (10.0.0.1)
       0.297ms 0.257ms 0.174ms
2 adsl-65.myprovider.be (217.136.111.1)
       12.120ms 13.058ms 13.009ms
3 194.78.255.177 (194.78.255.177)
       13.845ms 14.308ms 12.756ms
4 gigabitethernet2-2.intl2.gam.brussels.skynet.be (195.238.2.226)
       13.123ms 13.164ms 12.527ms
5 pecbru2.car.belbone.be (194.78.255.118)
       16.336ms 13.889ms 13.028ms
6 ser-2-1-110-ias-be-vil-ar01.kpnbelgium.be (194.119.224.9)
       14.602ms 15.546ms 15.959ms
7 unknown-195-207-939.eunet.be (195.207.93.49)
       16.514ms 17.661ms 18.889ms
8 S0-1-0.Leuven.Belgium.EU.net (195.207.129.1)
       22.714ms 19.193ms 18.432ms
9 dukat.Belgium.EU.net (193.74.208.178) 22.758ms * 25.263ms
```

In alcuni sistemi **traceroute** è stato rinominato **tracepath**.

10.2.6.4. Il comando whois

Si possono richiedere specifiche informazioni sul nome del dominio usando il comando **whois**, come viene spiegato da molti server **whois**, come quello che segue:

```
[emmy@pc10 emmy]$ whois cnn.com
[whois.crsnic.net]

Whois Server Version 1.3

        $<--snap server message-->

   Domain Name: CNN.COM
   Registrar: NETWORK SOLUTIONS, INC.
   Whois Server: whois.networksolutions.com
   Referral URL: http://www.networksolutions.com
   Name Server: TWDNS-01.NS.AOL.COM
   Name Server: TWDNS-02.NS.AOL.COM
   Name Server: TWDNS-03.NS.AOL.COM
   Name Server: TWDNS-04.NS.AOL.COM
   Updated Date: 12-mar-2002
>>> Last update of whois database: Fri, 5 Apr 2002 05:04:55 EST <<<

The Registry database contains ONLY .COM, .NET, .ORG, .EDU domains
and Registrars.

[whois.networksolutions.com]

        $<--snap server message-->

Registrant:
Turner Broadcasting (CNN-DOM)
   1 CNN Center
   Atlanta, GA 30303

Domain Name: CNN.COM
Administrative Contact:
     $<--snap contactinfo-->
Technical Contact:
     $<--snap contactinfo-->
Billing Contact:
     $<--snap contactinfo-->
Record last updated on 12-Mar-2002.
Record expires on 23-Sep-2009.
Record created on 22-Sep-1993.
Database last updated on 4-Apr-2002 20:10:00 EST.

Domain servers in listed order:

TWDNS-01.NS.AOL.COM          149.174.213.151
TWDNS-02.NS.AOL.COM          152.163.239.216
TWDNS-03.NS.AOL.COM          205.188.146.88
TWDNS-04.NS.AOL.COM          64.12.147.120
```

Per altri nomi di domini al di fuori di .com, .net, .org e .edu, potreste aver bisogno di specificare il server di whois, come in questo caso per i domini .be:

whois ***domain.be@whois.dns.be***

10.3. Applicazioni Internet/Intranet

Il sistema Linux è una grande piattaforma per offrire servizi di rete. In questa sezione proveremo ad offrire una panoramica dei server e delle applicazioni di rete più comuni.

10.3.1. Tipi di server

10.3.1.1. Server indipendente

L'offerta di un servizio agli utenti può essere affrontata in due modi: un demone o servizio può girare in modo indipendente o può dipendere da un altro servizio da attivare.

I servizi di rete che vengono usati pesantemente e/o in continuazione, girano normalmente in modo indipendente: sono dei programmi demoni indipendenti che stanno sempre in attività. Molto facilmente vengono fatti partire al momento dell'avvio del sistema ed attendono richieste su specifici punti o porte di connessione che essi sono stati impostati ad ascoltare. Quando giunge una richiesta, questa viene elaborata e l'ascolto continua fino alla successiva. Un server web è un esempio tipico: volete che sia disponibile 24 ore al giorno e, se è troppo occupato, dovrebbe creare ulteriori istanze di ascolto in modo da servire utenti simultanei. Altri esempi sono i grossi archivi di software come Sourceforge o il vostro mirror Tucows che devono gestire migliaia di richieste FTP al giorno.

Un esempio di servizio di rete indipendente nel vostro computer casalingo potrebbe essere **named** (*name daemon*), un server dei nomi con cache. I servizi indipendenti hanno i loro processi attivi, che voi potete controllare ogni volta usando **ps**:

```
bob:~> ps auxw | grep named
named   908 0.0 1.0 14876 5108 ? S Mar14 0:07 named -u named
```

Comunque esistono alcuni servizi che potete usare nel vostro PC anche se non c'è un processo server attivo per essi. Esempi possono essere il servizio FTP, il servizio di copia sicura o quello di finger; Tali servizi hanno il Demone Internet (**inetd**) in ascolto al loro posto.

10.3.1.2. (x)inetd

Di solito nel vostro PC domestico le faccende sono un po' più tranquille. Potete avere una piccola rete, per esempio, e dei file da trasferire da un PC ad un altro di tanto in tanto usando FTP o Samba (per il collegamento a macchine MS Windows). In tali casi avviare tutti i servizi di cui avete bisogno solo saltuariamente e tenerli tutto il tempo in funzione, sarebbe uno spreco di risorse. Così nelle configurazioni minori scoprirete che i demoni necessari dipendono da un programma centrale che ascolta su tutte le porte dei servizi di cui è responsabile.

Questo superserver, il demone dei servizi Internet, viene avviato durante l'inizializzazione del sistema. Esistono due diffuse implementazioni: **inetd** e **xinetd** (extended Internet services daemon). L'uno o l'altro girano di solito in ogni sistema Linux:

```
bob:~> ps -ef | grep inet
root 926    1 0 Mar14 ?   00:00:00 xinetd-ipv6 -stayalive -reuse \
-pidfile /var/run/xinetd.pid
```

I servizi di cui è responsabile il demone Internet sono elencati nel suo file di configurazione `/etc/inetd.conf` per **inetd** e nella directory `/etc/xinetd.d` per **xinetd**. Di consueto i

servizi gestiti comprendono la condivisione dei file e i quelli di stampa, SSH, FTP, telnet, il demone di configurazione Samba, i servizi del parlato e degli orari.

Non appena viene ricevuta una richiesta di connessione, il server centrale avvierà un'istanza del server desiderato. In questo modo, nell'esempio seguente, quando l'utente *bob* avvia una sessione FTP nell'host locale, parte un demone FTP non appena la sessione è attiva:

```
bob:~> ps auxw | grep ftp
bob     793 0.1 0.2 3960 1076 pts/6       S    16:44   0:00 ncftp localhost
ftp     794 0.7 0.5 5588 2608 ?           SN   16:44   0:00 ftpd:
localhost.localdomain: anonymous/bob@his.server.com: IDLE
```

Naturalmente succede la stessa cosa quando aprite delle connessioni verso host remoti: o un demone risponde direttamente, oppure un **(x)inetd** remoto avvia il servizio che vi serve e lo ferma quando avete finito.

10.3.2. La posta

10.3.2.1. I server

Sendmail è il programma server standard di posta o Mail Transport Agent per le piattaforme UNIX. E' robusto, scalabile e, se configurato adeguatamente per l'hardware corretto, gestisce migliaia di utenti senza battere ciglio. Maggiori informazioni su come configurare Sendmail sono incluse nei pacchetti sendmail e sendmail-cf: potreste leggere i file `README` e `README.cf` in `/usr/share/doc/sendmail`. Sono utili anche **man *sendmail*** e **man *aliases***.

Qmail è un altro server di posta che sta guadagnando in popolarità perché si vanta di essere più sicuro di Sendmail. Mentre Sendmail è un programma monolitico, Qmail è formato da parti più piccole di programma interagenti, che possono essere resi maggiormente sicuri. Postfix è un altro server di posta di crescente popolarità.

Questi server gestiscono mailing list, filtri, scansioni antivirus e molto altro ancora. Programmi liberi o commerciali di scansione antivirus sono disponibili per l'uso con Linux. Esempi di software per mailing list sono Mailman, Listserv, Majordomo e EZmlm. Guardate le pagine web del vostro scanner antivirus preferito per informazioni sul supporto dei clienti e server Linux. Amavis e Spamassassin sono implementazioni libere di virus scanner e spam scanner.

10.3.2.2. Server remoti di posta

I protocolli più diffusi per accedere alla posta in remoto sono *POP3* e *IMAP4*. IMAP e POP permettono entrambi operazioni non in linea, accesso remoto alla posta in arrivo e si appoggiano su un server SMTP per inviare la posta.

Mentre POP è un protocollo semplice, facile da implementare e supportato da quasi tutti i clienti di posta, IMAP è preferibile in quanto:

- può manipolare i flag persistenti dello stato dei messaggi;
- può sia conservare al meglio i messaggi di posta così come ridistribuirli;

- può utilizzare e gestire caselle di posta multiple;
- supporta aggiornamenti concorrenti e caselle postali condivise;
- è pure utilizzabile per accedere ai messaggi di Usenet e ad altri documenti;
- IMAP funziona sia online che offline;
- è ottimizzato per le prestazione online, soprattutto nei collegamenti a velocità estremamente bassa.

10.3.2.3. Mail user-agent

Esistono clienti di posta elettronica, sia testuali che grafici, in abbondanza: vi citeremo il nome di alcuni tra i più diffusi. Scegliete il vostro preferito.

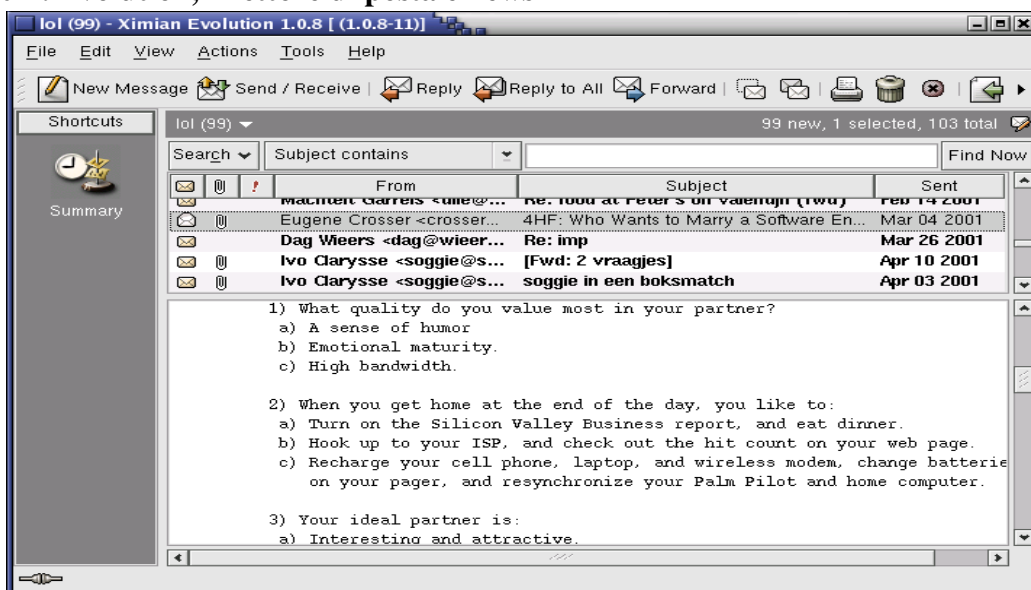
Il comando UNIX **mail** è stato in circolazione per molti anni, anche prima che esistessero le reti. E' una semplice interfaccia per mandare messaggi e piccoli file ad altri utenti, che può poi salvare il messaggio, redirigerlo, replicare ad esso e così via.

Mentre solitamente non è più usato come cliente, il programma **mail** è ancora utile, ad esempio, per inviare a qualcuno i dati in uscita di un programma:

mail `<future.employer@whereIwant2work.com> < cv.txt`

Il lettore di posta **elm** è un miglioramento molto più raffinato di **mail**, e così anche **pine** (Pine Is Not Elm). Il lettore di posta **mutt** è ancora più recente e offre funzioni quali il threading.

Per quegli utenti che preferiscono un'interfaccia grafica per la propria posta (e un gomito da tennista o un braccio da mouse), esistono centinaia di possibilità. I più popolari tra i nuovi utenti sono Mozilla Mail/Thnderbird, che ha delle facili opzioni di configurazione anti-spam, ed Evolution, il clone di MS Outlook. Kmail è popolare tra gli utenti KDE.

Figura 10-1. Evolution, il lettore di posta e news

Sono disponibili anche decine di applicazioni di posta web, come Squirrelmail, Yahoo!mail, gmail di Google e Hotmail.

Una panoramica si trova in Linux Mail User HOWTO.

Molte distribuzioni Linux includono **fetchmail**, un programma di utilità per la ricerca e l'inoltro della posta. Ritira la posta da server remoti di posta (POP, IMAP ed alcuni altri) e la inoltra al vostro sistema locale di consegna. Allora potete gestire la posta ricevuta attraverso normali clienti di posta. Può essere avviato in modalità demone per interrogare ripetutamente uno o più sistemi ad intervalli specifici. Informazioni ed esempi di impiego sono rintracciabili sulle pagine Info: la directory `/usr/share/doc/fetchmail[-<versione>]` contiene un elenco completo delle caratteristiche e una FAQ per i principianti.

Il filtro di **procmail** si può usare per filtrare la posta in entrata, per creare mailing list, per preprocessare la posta, per inoltrare selettivamente la posta ed altro ancora. L'unito programma **formail**, fra le altre cose, abilita alla generazione di risposte automatiche e la suddivisione in caselle postali. Procmail si trova ormai da anni nelle macchine UNIX e Linux ed è un sistema molto robusto, progettato per funzionare anche nelle circostanze peggiori. Maggiori informazioni si possono trovare nella directory `/usr/share/doc/procmail[-<versione>]` e nelle pagine man.

Una nota sull'Etichetta della Posta Elettronica
Alcune persone oggigiorno sembrano ritenere che un messaggio di posta elettronica non debba essere troppo formale. Ciò dipende, naturalmente. Se scrivete a qualcuno che non conoscete, è meglio mantenere un po' le distanze, proprio come fareste in una lettera tradizionale. E non dimenticate: le persone che non conoscete potrebbero essere maschi o femmine...

10.3.3. Il web

10.3.3.1. Il server web Apache

Apache è di gran lunga il più diffuso server web, impiegato in oltre la metà di tutti i server web di Internet. La maggioranza delle distribuzioni Linux includono Apache. I pregi di Apache comprendono il suo design modulare, il supporto SSL, stabilità e velocità. Con gli appropriati hardware e configurazioni può sostenere i carichi più grossi.

Nei sistemi Linux la configurazione del server si esegue solitamente nella directory `/etc/httpd`. Il principale file di configurazione è `httpd.conf`: è notevolmente autoesplicante. Se doveste aver bisogno di aiuto, lo potreste trovare nella pagina man di **httpd** nel sito web di Apache.

10.3.3.2. Navigatori del Web

Per la piattaforma Linux esiste un certo numero di navigatori web, sia gratuiti che commerciali. Così come Netscape Navigator è stato a lungo l'unica scelta decente per il passato, ora Mozilla/Firefox offre una competitiva alternativa che gira sotto molti altri sistemi operativi, come MS Windows ed anche MacOS X.

Amaya è il navigatore di W3C. Opera è un navigatore commerciale, compatto e rapido. Molti gestori di desktop offrono funzioni di navigazione web nei loro gestori di file, come **nautilus**.

Fra i diffusi navigatori basati sul testo ci sono **lynk** e **links**. Potreste avere necessità di installare dei server proxy nella vostra shell impostando le variabili giuste. I navigatori testuali sono veloci e maneggevoli quando non è disponibile un ambiente grafico, come quando si usano negli script.

10.3.3.3. I server proxy

10.3.3.3.1. Cos'è un server proxy?

Le società e le organizzazioni spesso esigono che i propri utenti utilizzino un server. Specialmente in ambienti con molti utenti, un server proxy può rendere più rapidi gli scaricamenti delle pagine web. Il server proxy conserva le pagine web. Quando un utente richiama una pagina web già richiesta precedentemente, il server proxy gli fornisce la pagina immediatamente, cosicché non ha necessità di ottenerla tramite Internet, cosa che richiederebbe più tempo. Naturalmente essere adottate delle misure in modo che il server proxy esegua un rapido controllo e fornisca sempre la versione più recente della pagina. In alcuni ambiti l'uso del server proxy è obbligatorio, in altri potete decidere se usarlo o meno.

10.3.3.3.2. La configurazione del proxy

Se avete il nome e la porta del server proxy, sarebbe piuttosto ovvio dare tale informazione al vostro navigatore. Tuttavia, molte applicazioni (a linea di comando) dipendono dalle variabili `http_proxy` e `ftp_proxy` per il corretto funzionamento. Per vostra comodità potreste voler aggiungere al vostro `~/.bashrc` una linea come quella che segue:

export **`http_proxy=`***`http://username:password@proxy_server_name:port_number`*

Per esempio:

export **`http_proxy=`***`http://willy:Appelsi3ntj3@proxy:80`*

Se non vi serve dare un nome utente e una password, escludete semplicemente ogni cosa prima del segno “@” compreso.

10.3.4. File Transfer Protocol

10.3.4.1. I server FTP

In un sistema Linux un server FTP viene normalmente avviato da **xinetd** usando il server *WU-ftpd*, sebbene il server FTP possa essere configurato come server indipendente in sistemi con traffico FTP pesante. Guardate gli esercizi.

I server FTP comprendono, tra gli altri, vsftpd, Ncftpd e Proftpd.

La maggior parte delle distribuzioni Linux contengono il pacchetto *anonftp*, che predispone un albero di server FTP anonimo ed i relativi file di configurazione.

10.3.4.2. I clienti FTP

La maggioranza delle distribuzioni Linux comprendono **ncftp**, una versione migliorata del comando UNIX **ftp**, che potreste conoscere anche attraverso la linea di comando Windows. Il programma **ncftp** offre funzionalità extra, come una più bella e comprensibile interfaccia utente, il completamento dei nomi dei file, le funzioni di aggiunta (*append*) e ripresa (*resume*), i segnalibri, la gestione della sessione ed altro ancora:

```
thomas:~> ncftp blob
NcFTP 3.0.3 (April 15, 2001) by Mike Gleason (ncftp@ncftp.com).
Connecting to blob...
blob.some.net FTP server (Version wu-2.6.1-20) ready.
Logging in...
Guest login ok, access restrictions apply.
Logged in to blob.
ncftp / > help

Commands may be abbreviated. 'help showall' shows hidden and
unsupported commands.
'help <command>' gives a brief description of <command>.

ascii      cat      help     lpage    open     quote   site
bgget      cd       jobs     lpwd     page     rename  type
bgput      chmod    lcd      lrename  pdir     rhelp   umask
bgstart    close    lchmod   lrm      pls      rm      version
binary     debug    lls      lrmdir   put      rmdir
bookmark   dir      lmkdir   ls       pwd      set
bookmarks  get      lookup   mkdir    quit     show
ncftp / >
```

Aiuti eccellenti con molti esempi si possono trovare nelle pagine man. E, di nuovo, è disponibile un certo numero di applicazioni GUI.

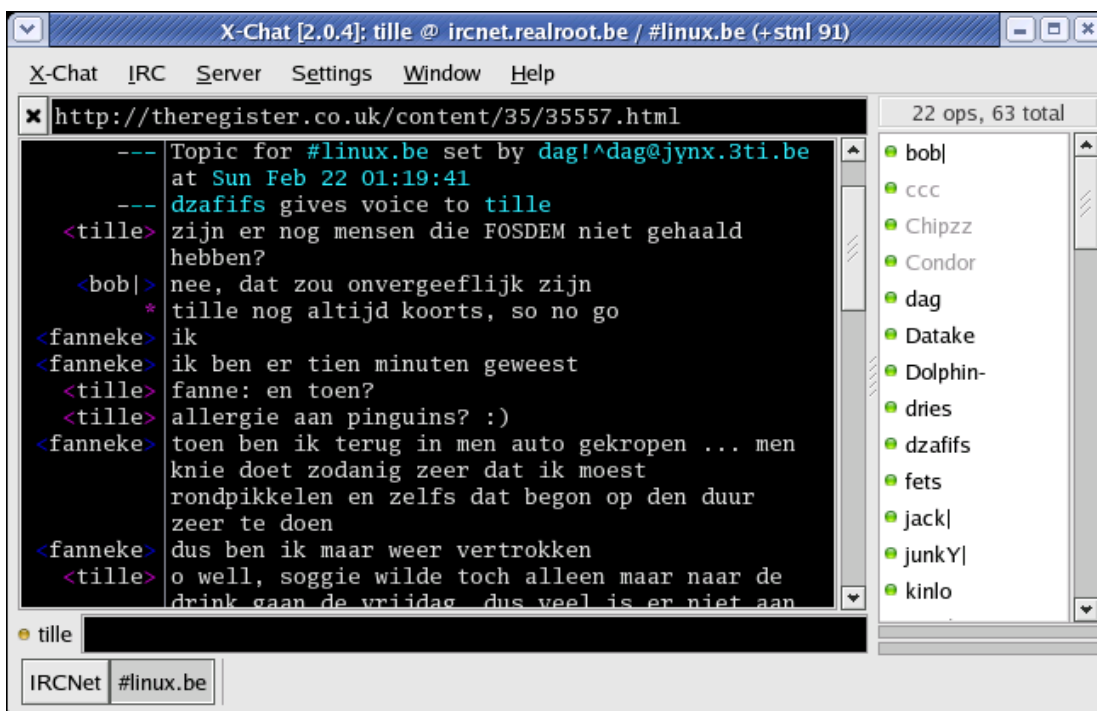
FTP non è sicuro!

Non usate il File Transfer Protocol per accessi non anonimi a meno che non sappiate cosa state facendo. I vostri nomi utente e password potrebbero essere captati da malevoli utenti compagni di rete! Usate invece FTP sicuro: il programma **sftp** è compreso nella suite Secure SHell (guardate la Sezione 10.4.4.).

10.3.5. Chat e conferenze

Vari clienti e sistemi sono a disposizione in ogni distribuzione in sostituzione della chat vecchio stile IRC, basata sul testo. Un breve ed incompleto elenco dei più diffusi programmi:

- **gaim**: cliente multiprotocollo di messaggistica istantanea per Linux, Windows e Mac, compatibile con MSN Messenger, ICQ, IRC e molti altri ancora: guardate le pagine Info o il sito di Gaim per maggiori informazioni.
- **xchat**: cliente IRC per il sistema X window:

  Figura 10-2. X-Chat

La pagina iniziale è su Sourceforge.

- **aMSN**: clone di MSN.
- **Konversation**, **kopete**, **KVIrc** e molti altri strumenti K dalla suite KDE.
- **gnomemeeting**: programma di videoconferenza per UNIX (ora Ekiga).
- **jabber**: piattaforma open source di messaggistica istantanea compatibile con ICQ, AIM, Yahoo, MSN, IRC, SMTP e molti altri ancora.
- **psi**: cliente jabber, guardate la pagina iniziale di PSI Jabber Client.
- **skype**: programma per effettuare chiamate in stile telefonico tramite Internet verso altri utenti Skype (guardate http://www.skype.com per maggiori informazioni). Skype è gratuito ma non aperto.
- **Gizmo**: un telefono gratuito (ma non aperto) per il vostro computer (guardate http://www.gizmoproject.com).

10.3.6. Servizi per notizie

L'avvio di un server Usenet implica molte prove e una regolazione fine, perciò fate riferimento alla pagina iniziale INN per maggiori informazioni.

Esiste una coppia di interessanti newsgroup nella gerarchia *comp.**, che può essere raggiunta con svariati clienti in modalità testo e grafica. Molti clienti di posta supportano bene la navigazione nei newsgroup: controllate il vostro programma o cercate nel vostro mirror di software a sorgente aperto dei clienti testuali tipo **tin**, **slrnn** e **mutt**, oppure scaricate Mozilla o uno degli altri numerosi clienti in modalità grafica.

Deja.com mantiene un archivio con ricerca di tutti i newsgroup, motorizzato da Google. Questo è uno strumento estremamente potente per ricevere aiuto: sono molto alte le probabilità che qualcuno

abbia già avuto i vostri problemi, abbia trovato una soluzione, poi postata in uno dei newsgroup.

10.3.7. Il Domain Name System

Tutte queste applicazioni necessitano dei servizi DNS per far corrispondere gli indirizzi IP ai nomi degli host e viceversa. Un server DNS non conosce tutti gli indirizzi IP del mondo, ma è collegato in rete con altri server DNS che può interrogare per trovare un indirizzo sconosciuto. La maggioranza dei sistemi UNIX possono far girare **named**, che fa parte del pacchetto BIND (Berkeley Internet Name Domain) distribuito da Internet Software Consortium. Può funzionare come *server dei nomi* indipendente con memorizzazione, cosa che fanno spesso i sistemi Linux per accelerare l'accesso in rete.

Il vostro principale file di configurazione è `/etc/resolv.conf`, che stabilisce l'ordine in cui vengono contattati i Domain Name Server:

```
search somewhere.org
nameserver 192.168.42.1
nameserver 193.74.208.137
```

Maggiori informazioni si possono trovare nelle pagine Info su **named**, nei file `/usr/share/doc/bind[-<versione>]` e nella pagina iniziale del progetto Bind. Il DNS HOWTO tratta dell'uso di BIND come DNS server.

10.3.8. DHCP

DHCP è il Dynamic Host Configuration Protocol che sta gradualmente rimpiazzando il buon vecchio **bootp** negli ambiti più grandi. Viene usato per controllare parametri vitali di rete quali gli indirizzi IP e i server dei nomi degli host. DHCP è compatibile a ritroso con **bootp**. Per configurare il server avrete bisogno di leggere l'HOWTO.

Le macchine clienti DHCP normalmente saranno configurate usando una GUI che imposta **dhcpd**, il demone cliente DHCP. Verificate la documentazione del vostro sistema se avete necessità di configurare il computer come cliente DHCP.

10.3.9. Servizi di autenticazione

10.3.9.1. La tradizione

Per tradizione gli utenti vengono autenticati localmente usando le informazioni conservate in `/etc/passwd` e `/etc/shadow` in ogni sistema. Ma anche quando si usa un servizio di rete per l'autenticazione, i file locali saranno sempre presenti per configurare gli account di sistema ad uso amministrativo, come l'account di root, gli account dei demoni e spesso quelli per programmi e scopi addizionali.

Questi file spesso sono i candidati principali per essere esaminati dagli hacker, perciò accertatevi che i permessi e le appartenenze siano rigidamente impostate in modo opportuno:

```
bob:~> ls -l /etc/passwd /etc/shadow
-rw-r--r--   1 root     root        1803 Mar 10 13:08 /etc/passwd
-r--------   1 root     root        1116 Mar 10 13:08 /etc/shadow
```

10.3.9.2. PAM

Linux può usare PAM, Pluggable Authentication Module, un metodo flessibile per l'autenticazione in UNIX. Vantaggi di PAM:

- Un comune schema di autenticazione che può essere utilizzato da un'ampia varietà di applicazioni.
- PAM può essere implementato con varie applicazioni senza la necessità di dover ricompilarle per supportare PAM specificamente.
- Grandi flessibilità e controllo sull'autenticazione per l'amministratore e lo sviluppatore di applicazioni.
- Gli sviluppatori di applicazioni non devono sviluppare i loro programmi per utilizzare uno specifico schema di autenticazione, mentre possono concentrarsi solo sui dettagli dei loro programmi.

La directory `/etc/pam.d` contiene i file di configurazione di PAM (di solito `/etc/pam.conf`). Ogni applicazione o servizio ha il suo file. Ciascuna linea nel file ha quattro elementi:

- *Modulo*:

    - `auth`: fornisce la vera autenticazione (forse richiedendo e verificando una password) ed imposta le credenziali, come l'appartenenza ad un gruppo o i permessi di Kerberos.
    - `account`: verifica per essere sicuri che l'accesso sia consentito all'utente (l'account non è scaduto, l'utente ha il permesso di collegarsi a quell'ora del giorno, e così via).
    - `password`: usato per impostare le password.
    - `session`: utilizzato dopo che un utente è stato autenticato. Questo modulo svolge compiti aggiuntivi che sono necessari a consentire l'accesso (per esempio, il montaggio della directory personale dell'utente o la messa a disposizione della sua casella di posta).

  L'ordine in cui i moduli in cui sono eseguiti, dal momento che ne vengono usati molti, è piuttosto importante.

- *Flag di controllo*: dicono a PAM quali azioni intraprendere in caso di accesso negato o consentito. I valori possono essere `required`, `requisite`, `sufficient` o `optional`.

- *Percorso del modulo*: percorso del modulo inseribile che deve essere usato, solitamente in `/lib/security`.

- *Argomenti*: informazioni per i moduli.

I file delle shadow password vengono individuati automaticamente da PAM.

Maggiori informazioni si possono trovare nelle pagine man di **pam** o nella pagina iniziale del progetto Linux-PAM.

10.3.9.3. LDAP

Il Lightweight Directory Access Protocol è un sistema cliente-server per accedere a servizi di directory globali o locali in una rete. In Linux si usa l'implementazione OpenLDAP. Essa comprende **slapd**, un server indipendente, **slurpd**, un server indipendente di replica LDAP, librerie che implementano il protocollo LDAP e una serie di programmi di utilità, strumenti e clienti di esempio.

Il maggior beneficio nell'uso di LDAP è il consolidamento di certi tipi di informazioni all'interno della vostra organizzazione. Per esempio tutti i differenti elenchi di utenti nella vostra organizzazione possono essere fusi in una sola directory LDAP. Questa directory può ricevere richieste da qualsiasi applicazione abilitata a LDAP che abbia bisogno di tale informazione. Possono accedere ad essa anche utenti che hanno bisogno delle informazioni della stessa.

Altri benefici di LDAP o X.500 Lite comprendono la sua facilità di installazione (rispetto a X.500) e la sua Interfaccia di Programmazione delle Applicazioni (API o Application Programming Interface), che significa che il numero delle applicazioni e dei gateway con LDAP dovrebbe aumentare in futuro.

Per quanto riguarda gli aspetti negativi, se volete usare LDAP, avete bisogno di applicazioni abilitate a LDAP oppure la capacità di utilizzare gateway LDAP. Mentre l'impiego di LDAP dovrebbe solo aumentare, attualmente non esistono molte applicazioni abilitate disponibili per Linux. Inoltre, mentre LDAP supporta qualche controllo degli accessi, non possiede tutte quelle caratteristiche di sicurezza di X.500.

Dal momento che LDAP è un protocollo aperto e configurabile, può essere impiegato per conservare ogni tipo di informazione relativa ad una struttura organizzativa particolare. Esempi comuni sono i lookup degli indirizzi di posta, l'autenticazione accentrata in combinazione con PAM, gli elenchi telefonici e i database di configurazione delle macchine.

Guardate le informazioni specifiche del vostro sistema e le pagine man per i relativi comandi come **ldapmodify** e **ldapsearch** per i dettagli. Maggiori informazioni si possono trovare in LDAP Linux HOWTO, che tratta dell'installazione, della configurazione, del funzionamento e della manutenzione di un server LDAP in Linux. LDAP Implementation HOWTO descrive gli aspetti tecnici della conservazione dei dati di applicazioni in un server LDAP sotto Linux. L'autrice di questo testo "Introduzione a Linux" ha pure scritto LDAP Operations HOWTO, che illustra gli elementi basilari che ognuno dovrebbe conoscere quando ha a che fare con la gestione, le operazioni e l'integrazione dei servizi di LDAP.

10.4. Esecuzione remota di applicazioni.

10.4.1. Introduzione

Ci sono due modi differenti per eseguire comandi o per avviare programmi in una macchina remota ed ottenere dei dati in uscita, testuali o grafici, mandati alle vostre workstation. Le connessioni possono essere sicure o meno. Sebbene naturalmente sia consigliabile usare connessioni sicure invece di inviare in rete la vostra password non criptata, parleremo di alcune applicazioni pratiche dei sistemi più vecchi (insicuri) dal momento che sono ancora utili in un moderno ambiente di rete, come, ad esempio, per l'individuazione dei problemi o per il funzionamento di esotici programmi.

10.4.2. Rsh, rlogin e telnet

I comandi **rlogin** e **rsh** per la connessione e l'esecuzione di comandi in remoto sono derivati da UNIX. Quantunque raramente usati perché palesemente insicuri, essi vengono ancora forniti con quasi tutte le distribuzioni Linux per compatibilità all'indietro con i programmi UNIX.

D'altro canto telnet viene tuttora comunemente utilizzato, spesso da amministratori di sistemi e di rete. Telnet è uno dei più potenti strumenti per l'accesso da remoto ai file e per l'amministrazione remota, consentendo connessioni da ovunque su Internet. Combinandolo con un server X le applicazioni grafiche da remoto possono essere mostrate localmente. Non esiste differenza tra il lavoro su macchine locali e l'uso di macchine remote.

Poiché l'intero collegamento non è criptato, permettere connessioni con **telnet** implica l'assunzione di notevoli rischi per la sicurezza. Per la normale esecuzione remota di programmi è consigliato Secure SHell o **ssh**. Parleremo del metodo sicuro più avanti in questa sezione.

Comunque **telnet** viene ancora utilizzato in molti casi. Qui di seguito ci sono alcuni esempi in cui un server di posta e un server web vengono provati per le risposte:

controllo del funzionamento di un server di posta:

```
[jimmy@blob ~] telnet mailserver 25
Trying 192.168.42.1...
Connected to mailserver.
Escape character is '^]'.
220 m1.some.net ESMTP Sendmail 8.11.6/8.11.6; 200302281626
ehlo some.net
250-m1.some.net Hello blob.some.net [10.0.0.1], pleased to meet you
250-ENHANCEDSTATUSCODES
250-8BITMIME
250-SIZE
250-DSN
250-ONEX
250-ETRN
250-XUSR
250 HELP
mail from: jimmy@some.net
250 2.1.0 jimmy@some.net... Sender ok
rcpt to: davy@some.net
250 2.1.5 davy@some.net... Recipient ok
```

```
data
354 Enter mail, end with "." on a line by itself
test
.
250 2.0.0 g2MA1R619237 Message accepted for delivery
quit
221 2.0.0 m1.some.net closing connection
Connection closed by foreign host.
```

controllo che un server web risponda ad elementari richieste:

```
[jimmy@blob ~] telnet www.some.net 80
Trying 64.39.151.23...
Connected to www.some.net.
Escape character is '^]'.
HEAD / ;HTTP/1.1

HTTP/1.1 200 OK
Date: Fri, 22 Mar 2002 10:05:14 GMT
Server: Apache/1.3.22 (UNIX) (Red-Hat/Linux)
 mod_ssl/2.8.5 OpenSSL/0.9.6
 DAV/1.0.2 PHP/4.0.6 mod_perl/1.24_01
Last-Modified: Fri, 04 Jan 2002 08:21:00 GMT
ETag: "70061-68-3c3565ec"
Accept-Ranges: bytes
Content-Length: 104
Connection: close
Content-Type: text/html

Connection closed by foreign host.

[jimmy@blob ~]
```

Ciò è perfettamente sicuro perché non dovete dare mai un nome utente e/o una password per ottenere i dati voluti, cosicché nessuno può intercettare quelle importanti informazioni dalla rete.

10.4.3. Il sistema X Window

10.4.3.1. Funzioni di X

Come abbiamo già spiegato nel Capitolo 7 (v. Sezione 7.3.3), il sistema X Window arriva dotato di un server X che fornisce la grafica ai clienti che necessitano di immagini.

E' importante comprendere la distinzione tra il server X e le applicazioni clienti X. Il server X controlla lo schermo direttamente ed è responsabile di tutto l'input e output via tastiera, mouse e schermo. D'altro canto il cliente X non accede direttamente alle periferiche d'ingresso e d'uscita dei dati: esso comunica con il server X che gestisce l'input e l'output. E' il cliente X che svolge il lavoro vero, come il calcolo dei valori, l'avvio di applicazioni e così via. Il server X apre solo delle finestre per trattare l'ingresso e l'uscita dei dati per uno specifico cliente.

Nelle normali condizioni operative (modalità grafica) ogni workstation è un server X di se stesso, anche se fa girare solo delle applicazioni clienti. Tutte le applicazioni che state facendo funzionare (per esempio Gimp, una finestra di terminale, il vostro navigatore, la vostra applicazione da ufficio, lo strumento per suonare un CD, ecc...) sono clienti del vostro server X. Server e clienti in questo caso girano sulla stessa macchina.

Questa natura cliente/server del sistema X lo rende un ambiente ideale per l'esecuzione remota di applicazioni e programmi. Poiché in realtà il processo si sta svolgendo in una macchina remota, c'è bisogno di una scarsissima potenza della CPU nell'host locale. Tali macchine, agendo da server per X, vengono chiamate terminali X ed una volta erano molto diffuse. Maggiori informazioni possono essere rintracciate in Remote X applications mini-HOWTO.

10.4.3.2. Telnet e X

Se voleste usare **telnet** per mostrare delle applicazioni grafiche che stanno girando in una macchina remota, dovreste prima concedere l'accesso verso tale macchina al vostro schermo (al vostro server X!) con il comando **xhost**, digitando un comando simile a quello seguente in una finestra di terminale nella vostra macchina locale:

```
davy:~> xhost +remote.machine.com
```

Dopo di ciò, connettetevi all'host remoto e ditegli di mostrare la grafica nella macchina locale impostando la variabile ambientale `DISPLAY`:

```
[davy@remote ~] export DISPLAY="local.host.com:0.0"
```

Completato questo passo, qualsiasi applicazione avviata in questa finestra di terminale verrà mostrata nel vostro desktop, utilizzando le risorse remote per l'elaborazione, ad esclusione delle vostre risorse grafiche locali (il vostro sistema X).

Questa procedura suppone che voi abbiate un qualche tipo di server X (XFree86, X.org, Exceed, Cygwin) già impostato nella macchina dove volete mostrare le immagini. L'architettura e il sistema operativo della macchina cliente non sono importanti se non perché vi consentono di far girare un server X in essa.

Ricordate che anche mostrare una finestra di terminale dalla macchina remota è considerata come un'immagine.

10.4.4. La suite SSH

10.4.4.1. Introduzione

Ora la maggior parte dei sistemi UNIX e Linux fa girare Secure Shell per evitare i rischi di sicurezza derivanti da **telnet**. La maggioranza dei sistemi Linux usano una versione di OpenSSH, una implementazione Open Source del protocollo SSH, che fornisce sicure comunicazioni criptate fra host non certificati in una rete non certificata. Nell'impostazione standard le connessioni X vengono mandate automaticamente, ma anche delle porte TCP/IP arbitrarie possono essere inviate usando un canale sicuro.

Il cliente **ssh** si connette e si registra nel nome dell'host specificato. L'utente deve fornire la sua identità alla macchina remota come specificato nel file sshd_config, che normalmente può essere trovato in /etc/ssh. Il file di configurazione è piuttosto autoesplicante ed abilita di base le funzioni più comuni. Se avete bisogno di aiuto, lo potete trovare nelle pagine man di **sshd**.

Quando l'identità è stata accettata dal server, quest'ultimo o esegue il comando dato oppure registra nella macchina e concede all'utente una normale shell nella macchina remota. Tutte le comunicazioni con il comando o la shell remoti verranno automaticamente criptate.

La sessione termina quando il comando o la shell nella macchina remota si conclude e tutte le connessioni X11 e TCP/IP sono state chiuse.

Quando vi connettete per la prima volta ad un host usando uno qualsiasi dei programmi che sono inclusi nella collezione SSH, avete bisogno di stabilire l'autenticità di quell'host e di fargli comprendere che voi volete connettervi:

```
lenny ~> ssh blob
The authenticity of host 'blob (10.0.0.1)' can't be established.
RSA fingerprint is 18:30:50:46:ac:98:3c:93:1a:56:35:09:8d:97:e3:1d.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'blob,192.168.30.2' (RSA) to the list of
known hosts.
Last login: Sat Dec 28 13:29:19 2002 from octarine
This space for rent.

lenny is in ~
```

E' importante che battiate "yes", con tre caratteri, e non solo "y". Ciò edita il vostro file ~/.ssh/known_hosts (v. Sezione 10.4.4.3.).

Se volete solo controllare qualcosa in una macchina remota e poi ritornare al proprio prompt nell'host locale, potete dare i comandi che intendete eseguire in remoto come argomenti di **ssh**:

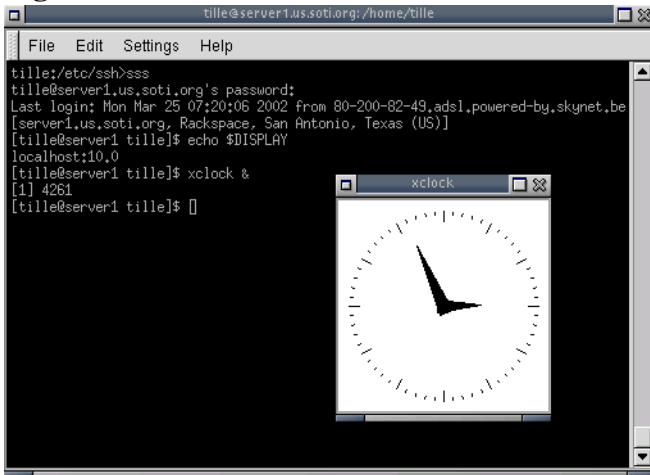
```
lenny ~> ssh blob who
jenny@blob's password:
root     tty2          Jul 24 07:19
lena     tty3          Jul 23 22:24
lena     0:            Jul 25 22:03

lenny ~> uname —n
magrat.example.com
```

10.4.4.2. Inoltro X11 e TCP

Se la voce `X11Forwarding` è impostata su *yes* nella macchina bersaglio e l'utente sta utilizzando delle applicazioni X, la variabile ambientale `DISPLAY` viene definita, la connessione allo schermo X11 viene inoltrata automaticamente verso il lato remoto in modo che qualsiasi programma X11 avviato dalla shell passerà attraverso il canale criptato e la connessione verso il vero server X sarà effettuata dalla macchina locale. L'utente non dovrebbe impostare manualmente `DISPLAY`. L'inoltro di connessioni X11 può essere configurato da linea di comando o nel file di configurazione **sshd**.

Il valore di `DISPLAY,` impostato da **ssh**, punterà alla macchina server, ma con un numero di schermo maggiore di zero. Ciò è normale e succede perché **ssh** crea un server X *proxy* nella macchina server (che fa girare le applicazioni clienti di X) per inoltrare le connessioni sul canale criptato.

Ciò è fatto completamente in automatico, così, quando battete il nome di un'applicazione grafica, essa viene mostrata nella vostra macchina locale e non nell'host remoto. Nell'esempio usiamo **xclock** dal momento che si tratta di un piccolo programma che viene solitamente installato ed è ideale per esperimenti:

Figura 10-3. Inoltro SSH X11

SSH imposterà anche i dati Xauthority automaticamente) nella macchina server. A tale scopo genererà un cookie casuale di autorizzazione, lo immagazzinerà in `Xauthority` nel server, controllerà che ogni connessione inoltrata trasporti questo cookie e lo rimpiazzerà con quello reale una volta aperta la connessione. Il vero cookie d autenticazione non è mai inviato alla macchina server (e nessun cookie viene inviato in chiaro).

L'inoltro di connessioni arbitrarie TCP/IP su un canale sicuro può essere specificato o da linea di comando o nel file di configurazione.

Il server X

Questa procedura suppone che abbiate un server X in funzione nel cliente in cui volete mostrare l'applicazione dal'host remoto. Anche se il cliente avesse architettura e sistema operativo diversi rispetto all'host remoto, purché quest'ultimo sia in grado di far girare un server X come Cygwin (che implementa un server X.org per clienti MS Windows e altri) o Exceed, dovrebbe essere possibile attivare una connessione remota con qualsiasi macchina Linux o UNIX.

10.4.4.3. Autenticazione del server

Il sistema cliente/server di **ssh** gestisce e controlla in modo automatico un database contenente gli identificativi di tutti gli host in cui è stato utilizzato. Le chiavi degli host sono conservate in `$HOME/.ssh/known_hosts` nella directory personale dell'utente. Inoltre il file `/etc/ssh/ssh_known_hosts` viene automaticamente controllato per ricercare host conosciuti. Qualsiasi nuovo host viene aggiunto in modo automatico al file dell'utente. Se l'identificativo di un host dovesse cambiare, **ssh** lo segnalerebbe e disabiliterebbe l'autenticazione della password per evitare che un "cavallo di Troia" si impossessi della password dell'utente. Un altro scopo di questo meccanismo è la prevenzione dagli attacchi "dell'uomo in mezzo" [man in the

middle] che altrimenti potrebbero essere utilizzati per aggirare il criptaggio. In ambiti dove è richiesta un'elevata sicurezza, **sshd** può anche essere configurato per prevenire la registrazione [login] in macchine le cui chiavi di host sono state cambiate o sono sconosciute.

10.4.4.4. Copia sicura in remoto

La suite SSH offre **scp** come alternativa sicura al comando **rcp** che veniva utilizzato quando esisteva solamente **rsh**. **scp** utilizza **ssh** per il trasferimento dei dati, usa la medesima autenticazione e fornisce la stessa sicurezza di **ssh**. Diversamente da **rcp**, **scp** richiederà password o passphrase se sono necessarie all'autenticazione:

```
lenny /var/tmp> scp Schedule.sdc.gz blob:/var/tmp/
lenny@blob's password:
Schedule.sdc.gz 100% |*****************************| 100 KB 00:00

lenny /var/tmp>
```

Ogni nome di file può contenere una descrizione dettagliata dell'host e dell'utente per indicare che il file sta per essere copiato in/da quell'host. Le copie tra due host remoti sono permesse. Guardate le pagine Info per maggiori informazioni.

Se invece volete usare un'interfaccia tipo FTP, utilizzate **sftp**:

```
lenny /var/tmp> sftp blob
Connecting to blob...
lenny@blob's password:

sftp> cd /var/tmp

sftp> get Sch*
Fetching /var/tmp/Schedule.sdc.gz to Schedule.sdc.gz

sftp> bye

lenny /var/tmp>
```

Copia sicura o le GUI di FTP
Non vi sentite ancora a vostro agio con la linea di comando? Provate le capacità di Konqueror per la copia sicura in remoto oppure installate Putty.

10.4.4.5. Chiavi di autenticazione

Il comando **ssh-keygen** genera, gestisce e converte le chiavi di autenticazione di **ssh**. Può creare chiavi RSA da usare con il protocollo SSH versione 1 e chiavi RSA o DSA da impiegare con il protocollo SSH versione 2.

Normalmente ogni utente che vuole usare SSH con autenticazione RSA o DSA lo fa girare una volta per creare la chiave di autenticazione in `$HOME/.ssh/identity`, `id_dsa` o `id_rsa`. Inoltre l'amministratore di sistema può usarlo per generare le chiavi di host per il sistema.

Di solito questo programma genera la chiave e richiede un file in cui conservare la chiave privata. La chiave pubblica viene tenuta in un file con lo stesso nome ma con l'aggiunta di *.pub*. Il

programma domanda anche una passphrase. La passphrase può essere vuota per indicare di non serve (le chiavi di host devono avere una passphrase vuota), oppure può essere una stringa di lunghezza arbitraria.

Non c'è modo di ricostruire una passphrase persa: se infatti questa viene perduta o dimenticata, è necessario generare una nuova chiave e copiarla tra le corrispondenti chiavi pubbliche.

Studieremo le chiavi SSH negli esercizi. Tutte le informazioni possono essere trovate nelle pagine man o Info.

10.4.5. VNC

VNC o *Virtual Network Computing* è in sostanza un sistema di visualizzazione remota che consente di vedere un ambiente desktop non solo nella macchina locale in cui sta funzionando, ma anche da qualsiasi punto di Internet e da un'ampia varietà di macchine e di architetture, compresi MS Windows e diverse distribuzioni UNIX. Potreste, ad esempio, far girare MS Word in una macchina dotata di Windows NT e mostrare l'output nel vostro desktop Linux. VNC fornisce sia server che clienti, cosicché funziona anche viceversa ed è perciò possibile usarlo per mostrare programmi Linux in clienti Windows. Probabilmente VNC è il modo più semplice per avere connessioni X su un PC. Le seguenti caratteristiche differenziano VNC da un normale server X o da implementazioni commerciali:

- Lo stato non viene conservato sul lato del visualizzatore: potete lasciare la vostra scrivania e riprendere da un'altra macchina, continuando da dove eravate rimasti. Quando avete in funzione un PC server X ed il PC si blocca o viene riavviato, tutte le applicazioni remote, da voi fatte funzionare, cesseranno. Con VNC rimarranno in funzione.
- E' piccolo e semplice, non avendo bisogno d'installazione, se serve può essere avviato da un dischetto.
- Indipendente dalla piattaforma con il cliente Java, gira praticamente su ogni sistema che supporti X.
- Condivisibile: un desktop può essere mostrato in parecchi visualizzatori.
- Libero.

Maggiori informazioni possono essere trovate nelle pagine man del cliente VNC (**man *vncviewer***) o nel sito web di VNC.

10.4.6. Il protocollo rdesktop

Per la gestione semplice degli host MS Windows, le recenti distribuzioni di Linux supportano il Remote Desktop Protocol (RDP) che è inserito nel cliente **rdesktop**. Il protocollo viene utilizzato da un certo numero di prodotti Microsoft, compresi Windows NT Terminal Server, Windows 2000 Server, Windows Xp e Windows 2003 Server.

Sorprendete i vostri amici (o la dirigenza) con la modalità a tutto schermo, molti tipi di conformazioni di tastiere e la modalità a singola applicazione, come se fosse una cosa vera. Il manuale **man *rdesktop*** fornisce ulteriori informazioni. La pagina iniziale del progetto è su

http://www.rdesktop.org/.

10.4.7. Cygwin

Cygwin fornisce sostanziali funzionalità UNIX sotto sistemi MS Windows: a parte gli strumenti a linea di comando UNIX e le applicazioni grafiche, può essere usato per mostrare un desktop Linux in una macchina MS Windows, utilizzando X in remoto. Dalla shell Bash di Cygwin, digitate il comando

/usr/X11R6/bin/XWin.exe -query ***nomevostramacchinalinux_o_IP***

La connessione è negata come preimpostazione. Dovete perciò cambiare la configurazione di X Display Manager (XDM) e possibilmente quella di X Font Server (XFS) per abilitare tale genere di connessione con cui otterrete una schermata di login sulla macchina remota. In base al vostro gestore di desktop (Gnome, KDE o altro) potreste pure dover cambiare qui alcune configurazioni.

Se non vi serve far apparire l'intero desktop, potete usare SSH con Cygwin, proprio come spiegato nella Sezione 10.4.4. senza tutta la confusione delle modifiche ai file di configurazione.

10.5. La sicurezza

10.5.1. Introduzione

Non appena un computer viene connesso alla rete, tutti i generi di abusi divengono possibili, sia questo basato su UNIX o su qualsiasi altro sistema. Ammettiamo che montagne di carta sono state sprecate su tale argomento e che ci potrebbe condurre troppo lontano per discutere in dettaglio il tema della sicurezza. C'è comunque un paio di cose abbastanza logiche che anche un utente principiante può fare per ottenere un sistema molto sicuro, poiché molte intrusioni sono il risultato di utenti ignoranti o incuranti.

Potreste chiedervi se tutto ciò vi riguarda, utilizzando il vostro computer a casa o al lavoro nel vostro ufficio su un desktop in un ambiente sufficientemente protetto. Tuttavia le domande che dovreste porvi dovrebbero essere piuttosto di questo tipo:

- Volete avere il controllo del vostro sistema?
- Volete partecipare (involontariamente) ad attività criminose?
- Volete che il vostro sistema sia utilizzato da qualcun altro?
- Volete correre il rischio di perdere la vostra connessione ad Internet?
- Volete ripristinare il vostro sistema ogni volta che è stato craccato?
- Volete correre il rischio di perdere dati personali o altrui?

Supponendo che non lo vogliate, vi elencheremo rapidamente i passi da intraprendere per rendere sicura la vostra macchina. Informazioni estese si possono trovare nel Linux Security HOWTO.

10.5.2. I servizi

L'obbiettivo è quello di avviare meno servizi possibili. Se il numero delle porte aperte al mondo esterno viene mantenuto al minimo, ciò è meglio per tenerle sotto osservazione. Se i servizi non possono essere spenti per la rete locale, tentate almeno di disabilitarli per le connessioni esterne.

Una regola empirica è quella che se non riconoscete un particolare servizio, probabilmente non ne avete comunque bisogno. Tenete pure in mente che non ha senso utilizzare certi servizi in Internet. Non fidatevi di ciò che *dovrebbe* essere fatto girare, verificate con il comando **netstat** quali servizi sono in ascolto e su quale porta TCP:

```
[elly@mars ~] netstat -l | grep tcp
tcp     0      0 *:32769                *:* LISTEN
tcp     0      0 *:32771                *:* LISTEN
tcp     0      0 *:printer              *:* LISTEN
tcp     0      0 *:kerberos_master      *:* LISTEN
tcp     0      0 *:sunrpc               *:* LISTEN
tcp     0      0 *:6001                 *:* LISTEN
tcp     0      0 *:785                  *:* LISTEN
tcp     0      0 localhost.localdom:smtp *:* LISTEN
tcp     0      0 *:ftp                  *:* LISTEN
tcp     0      0 *:ssh                  *:* LISTEN
tcp     0      0 ::1:x11-ssh-offset     *:* LISTEN
```

Cose da evitare:

- **exec**, **rlogin**, **rsh** e **telnet** tanto per stare dalla parte sicura.
- X11 nelle macchine server;
- niente lp se nessuna stampante è fisicamente collegata;
- se non ci sono host MS Windows in rete non occorre Samba;
- non permettete FTP a meno che non sia richiesto un server FTP;
- non permettete NFS e NIS su Internet, disabilitate tutti i relativi servizi in una installazione indipendente.
- Non avviate un MTA, se non siete veramente in un server di posta.
- ...

Fermate i servizi attivi usando il comando **chkconfig**, gli initscript o modificando i file di configurazione di **(x)inetd**.

10.5.3. Aggiornare con regolarità

Ciò che fa prosperare Linux è la sua capacità di adattarsi rapidamente a qualsiasi mutevole ambiente. Ma genera pure la possibilità che gli aggiornamenti di sicurezza siano stati rilasciati anche mentre state installando una nuova versione di marca, cosicché la prima cosa che dovreste fare (e che vale per quasi tutti i sistemi operativi che vi vengono in mente) dopo l'installazione è quella di procurarvi gli aggiornamenti al più presto possibile. Dopo di ciò aggiornate con regolarità *tutti* i pacchetti.

Alcuni aggiornamenti possono richiedere dei nuovi file di configurazione e i vecchi file dovrebbero

essere rimpiazzati. Controllate la documentazione ed assicuratevi che ogni cosa funzioni normalmente dopo l'aggiornamento.

La maggioranza delle distribuzioni Linux fornisce servizi di mailing list per annunci di aggiornamenti di sicurezza e strumenti per applicarli al sistema. In generale i problemi di sicurezza solo di Linux vengono segnalati, fra l'altro, su Linuxsecurity.com.

L'aggiornamento è un procedimento continuo, sicché dovrebbe essere un'abitudine quasi quotidiana.

10.5.4. I firewall e le politiche d'accesso

10.5.4.1. Che cos'è un firewall?

Nella precedente sezione abbiamo già menzionato le capacità di un firewall sotto Linux. Mentre l'amministrazione di un firewall è uno dei compiti del vostro amministratore di rete, voi dovreste conoscere un paio di cose sui firewall.

Firewall è un termine vago che può significare qualcosa che si comporta da barriera protettiva tra noi e il mondo esterno, in genere Internet. Un firewall può essere un sistema dedicato o una specifica applicazione che fornisce tale funzione, oppure può trattarsi di una combinazione di componenti, comprese varie combinazioni di hardware e software. I firewall sono realizzati in base a "regole" che vengono usate per definire cosa può entrare e/o uscire in un dato sistema o rete.

Dopo la disabilitazione dei servizi non necessari, ora vogliamo limitare i servizi rimasti per permettere solo il minimo richiesto di connessioni. Un buon esempio funziona da casa: soltanto la specifica connessione tra il vostro ufficio e la vostra abitazione dovrebbe essere permessa, mentre le connessioni da altre macchine su Internet dovrebbero essere bloccate.

10.5.4.2. I filtri dei pacchetti

La prima linea di difesa è un *filtro dei pacchetti* (packet filter), che può guardare dentro i pacchetti IP e prendere decisioni in base al contenuto. Il più comune è il pacchetto Netfilter, che fornisce il comando **iptables**, un filtro di pacchetti di nuova generazione per Linux.

Uno dei miglioramenti maggiormente degni di nota nei kernel più recenti è la funzione di *stateful inspection*, che non solo dice quello che c'è all'interno di un pacchetto, ma anche scopre se un pacchetto appartiene o è in relazione a una nuova connessione o a quella esistente.

Shorewall Firewall, o Shorewall in breve, è un programma di interfaccia alle funzionalità del firewall standard in Linux.

Maggiori informazioni possono essere trovate nella pagina del progetto Netfilter/iptables.

10.5.4.3. TCP wrapper

Il wrapping del TCP fornisce molti degli stessi risultati dei filtri di pacchetti, ma funziona in modo diverso. Il wrapper accetta realmente il tentativo di connessione, poi esamina i file di

configurazione e decide se accettare o respingere la richiesta di connessione. Esso controlla le connessioni a livello dell'applicazione piuttosto che a quello di rete.

I TCP wrapper sono tipicamente utilizzati con **xinetd** per fornire il nome dell'host e il controllo dell'accesso basato sull'indirizzo IP. In aggiunta, questi strumenti comprendono capacità di registrazione e di gestione dell'utilizzazione che sono facili da configurare.

I vantaggi dei TCP wrapper sono che il cliente che si sta connettendo è inconsapevole che vengono usati i wrapper e che essi operano separatamente dalle applicazioni che stanno proteggendo.

L'accesso basato sull'host viene controllato nei file `host.allow` e `host.deny`. Maggiori informazioni si possono trovare nei file di documentazione del TCP wrapper in `/usr/share/doc/tcp_wrappers[-<versione>/]` o `/usr/share/doc/tcp` e nelle pagine man dedicate ai file di controllo dell'accesso basato sull'host che contengono degli esempi.

10.5.4.4. I proxy

I proxy possono svolgere diversi compiti, di cui non tutti hanno molta attinenza con la sicurezza. Ma il fatto che essi facciano da intermediari li rende un buon posto per rafforzare le politiche di controllo degli accessi, limitare le connessioni dirette attraverso un firewall e controllare come la rete dietro il proxy si affaccia su Internet.

Solitamente in combinazione con un filtro di pacchetti (ma qualche volta tutto da soli) i proxy forniscono un livello extra di controllo. Maggiori informazioni si possono trovare nel Firewall HOWTO nel sito web di Squid.

10.5.4.5. L'accesso a singole applicazioni

Alcuni server possono avere le loro proprie caratteristiche di controllo degli accessi. Comuni esempi comprendono Samba, X Window, Bind, Apache e CUPS. Per ogni servizio che volete offrire controllate quali file di configurazione si devono usare.

10.5.4.6. I file di log

Se mai, il modo UNIX di registrare tutti i tipi di attività in tutti i generi di file conferma che "si sta facendo qualcosa". Naturalmente i file di registro dovrebbero essere controllati con regolarità, manualmente o automaticamente. I firewall e gli altri mezzi di controllo degli accessi tendono a creare notevoli quantità di file di log, cosicché il trucco è quello di provare e di registrare solo le attività anomale.

10.5.5. La scoperta delle intrusioni

I sistemi di scoperta delle intrusioni (IDS o Intrusion Detection System) sono progettati per intercettare ciò che potrebbe aver superato il firewall. Essi o possono essere stati ideati per catturare un attivo tentativo di effrazione in corso, oppure per individuare una riuscita effrazione dopo il fatto. Nell'ultimo caso è troppo tardi per impedire un qualche danno, ma almeno abbiamo una precoce consapevolezza di un problema. Esistono due tipologie base di IDS: quelli che proteggono le reti e quelli che proteggono singoli host.

Per gli IDS basati sugli host ciò viene fatto tramite programmi di utilità che tengono controllati i mutamenti nel file system. I file di sistema che in qualche modo sono cambiati, ma non avrebbero dovuto cambiare, sono una chiara manifestazione che qualcosa non va. Chiunque entri ed ottenga l'accesso di root farà, presumibilmente, delle modifiche al sistema da qualche parte. Questa è di solito la primissima cosa che viene fatta, o in modo che si possa successivamente ritornare attraverso una backdoor, o lanciando un attacco contro qualcun altro (nel qual caso si debbono cambiare o aggiungere file al sistema). Alcuni sistemi sono dotati del sistema di monitoraggio **tripwire**, documentato nel sito web del Progetto Open Source Tripwire.

La scoperta delle intrusioni in rete è gestita da un meccanismo che osserva tutti il traffico che supera il firewall (non attraverso gli scanner delle porte che segnalano le porte utilizzabili). Snort è un esempio Open Source di siffatti programmi. Whitehats.com mantiene un database di scoperta delle intrusioni aperto, arachNIDS.

10.5.6. Ulteriori spunti

Alcune cose in generale da tenere in mente:

- Non consentite login di root. Gli sviluppatori UNIX hanno creato oltre vent'anni fa **su** per una maggiore sicurezza.
- L'accesso diretto in qualità di root [amministratore] è sempre pericoloso e suscettibile di errori umani, sia permettendo l'autenticazione di root, sia usando il comando **su**. Piuttosto che usare **su**, è molto meglio ricorrere a **sudo** per eseguire solo il comando che il quale vi servono i permessi extra e ritornare subito dopo al vostro ambiente personale.
- Prendete seriamente le password: utilizzate quelle "shadow". Cambiatele spesso.
- Provate ad usare sempre SSH o SSL. Evitate **telnet**, clienti FTP, di posta e altri programmi clienti che inviano le password in chiaro sulla rete. La sicurezza non riguarda solo il vostro computer, ma anche la vostra password.
- Limitate le risorse usando **quota** e/o **ulimit**.
- La posta per root dovrebbe essere consegnata a (o almeno letta da) una persona reale.
- L'Istituto SANS ha più trucchi e suggerimenti, ordinati per distribuzione, con servizio di mailing list.
- Controllate la provenienza del nuovo software, scaricatelo da un luogo/sito fidato. Verificate i nuovi pacchetti prima dell'installazione.
- Quando utilizzate una connessione Internet non permanente, chiudetela non appena non vi serve più.
- Fate funzionare i servizi privati su porte diverse da quelle che si aspettano eventuali hacker.
- Comprendete il vostro sistema. Dopo un po' potrete quasi avvertire quando sta capitando qualcosa.

10.5.7. Sono stato attaccato dagli hacker?

Come potete dirlo? Questa è una lista di controllo di eventi sospetti:

- Porte misteriose aperte, strani processi.
- Utilità di sistema (normali comandi) che si comportano in modo strano.
- Problemi di autenticazione.
- Uso inspiegabile di larghezza di banda.
- File di registro danneggiati o persi, demone di syslog che si comporta in modo strano.
- Interfacce in modalità inconsuete.
- File di configurazione modificati inaspettatamente.
- Strane voci nei file dello storico di shell.
- File temporanei sconosciuti.

10.5.8. Ripristinare dopo un'intrusione

In breve, state calmi. Poi eseguite le seguenti azioni in questo ordine:

- Sconnettete la macchina dalla rete.
- Tentate di scoprire quanto più potete sul modo in cui la vostra sicurezza è stata violata.
- Fate una copia di sicurezza dei dati importanti non di sistema. Se possibile confrontate questi dati con le copie di sicurezza esistenti, fatte prima che il sistema fosse compromesso, per garantire l'integrità dei dati.
- Reinstallate il sistema.
- Usate nuove password.
- Ripristinate dai backup di sistema e dei dati.
- Applicate tutti gli aggiornamenti disponibili.
- Riesaminate il sistema: bloccate i servizi non necessari, controllate le regole del firewall e le altre politiche d'accesso.
- Riconnettete

10.6. Sommario

Linux e le reti vanno mano nella mano. Il kernel di Linux ha il supporto per tutti i protocolli di rete comuni e meno comuni. Gli strumenti standard per le reti UNIX sono in dotazione ad ogni distribuzione. Oltre a questi, la maggior parte delle distribuzioni mette a disposizione strumenti per una semplice installazione e gestione delle reti.

Linux è ben conosciuto come una stabile piattaforma per far girare vari servizi Internet e la quantità di software per Internet è infinita. Come UNIX, Linux può essere e amministrato al meglio da una postazione remota, usando una delle diverse soluzioni per l'esecuzione remota dei programmi.

Abbiamo brevemente toccato l'argomento della sicurezza. Linux è un sistema firewall ideale, leggero ed economico, ma che può essere impiegato in molteplici altre funzioni di rete come i router e i server proxy.

L'aumento della sicurezza nella rete viene principalmente ottenuto installando frequentemente gli aggiornamenti e con il senso comune.

Ecco qui una panoramica dei comandi relativi alla rete:

Tabella 10-2. Nuovi comandi nel capitolo 10: Reti

Comando	Significato
ftp	Trasferisce file verso un altro host (non sicuro).
host	Assume informazioni sugli host in rete.
ifconfig	Mostra le informazioni dell'indirizzo IP
ip	Mostra le informazioni dell'indirizzo IP
netstat	Mostra le informazioni di instradamento e le statistiche di rete.
ping	Invia richieste di risposta ad altri host.
rdesktop	Mostra un desktop MS Windows nel vostro sistema Linux.
route	Mostra le informazioni di instradamento.
scp	Copia sicura dei file da e verso altri host.
sftp	FTP sicuro per i file da e verso altri host.
ssh	Crea una connessione criptata verso un altro host.
ssh-keygen	Genera le chiavi di autenticazione per Secure SHell.
telnet	Crea una connessione non sicura verso altri host.
tracepath/traceroute	Riporta il percorso che un pacchetto segue verso un altro host.
whois	Ottiene informazioni circa un nome di dominio.
xclock	Applicazione dell'orologio X Window, utile per testare le visualizzazioni in remoto.
xhost	Strumento X Window per il controllo degli accessi.

10.7. Esercizi

10.7.1. Le reti in generale

- Visualizzate le informazioni di rete della vostra workstation: indirizzo IP, instradamenti, server dei nomi.
- Fingete che non sia disponibile il DNS. Cosa dovreste fare per raggiungere il computer del vicino senza digitare ogni volta l'indirizzo IP?
- Come potreste immagazzinare permanentemente le informazioni del proxy per un navigatore testuale come **links**?
- Quali server dei nomi gestiscono il dominio redhat.com?
- Inviate un messaggio di posta elettronica al vostro account locale. Provate due modi diversi per mandarlo e leggerlo. Come potete controllare che è arrivato veramente?
- La vostra macchina accetta connessioni anonime FTP? Come usereste il programma **ncftp** per autenticarvi con il nome utente e la password vostri?
- La vostra macchina sta facendo girare un server web? Se così non è, fate che lo sia. Controllate i file di registro!

10.7.2. Connessioni remote

- Dalla vostra workstation locale visualizzate un'applicazione grafica (come **xclock**) sullo schermo del vostro vicino. Dovranno essere configurati gli account necessari. Utilizzate una connessione sicura!
- Impostate le chiavi SSH in modo che possiate connettervi alla macchina del vostro vicino senza dover inserire una password.
- Fate una copia di sicurezza della vostra directory personale in `/var/tmp` o nel "server di backup" del vostro vicino usando **scp**. Archiviate e comprimete prima di trasferire i dati! Connettetevi all'host remoto usando **ssh**, spacchettate la copia di sicurezza e rimettete un file nella macchina originale ricorrendo a **sftp**.

10.7.3. La sicurezza

- Redigete un elenco delle porte aperte (in ascolto) nella vostra macchina.
- Supponendo che vogliate far funzionare un web server, quali servizi dovreste disattivare? Come potreste farlo?
- Installate gli aggiornamenti disponibili.
- Come potete vedere chi è connesso al vostro sistema?
- Create un processo ripetitivo che vi ricordi di cambiare la vostra password ogni mese e preferibilmente anche quella di *root*.

Capitolo 11. Suoni e video

Questo capitolo si occupa dei seguenti argomenti (brevemente, dal momento che il settore del suono e del video è piuttosto ampio):

- configurazione della scheda sonora;
- riproduzione e copia dei CD:
- riproduzione di file musicali;
- controllo del volume;
- video e televisione;
- registrazione dei suoni.

11.1. Le basi dell'audio

11.1.1. Installazione

Molto probabilmente il vostro sistema è già installato con i driver audio e la configurazione è stata eseguita al momento dell'installazione. Allo stesso modo, se voi aveste mai bisogno di sostituire il vostro hardware audio, la maggior parte dei sistemi fornisce strumenti che consentono una impostazione e configurazione semplice del dispositivo. Le attuali schede sonore plug-and-play più diffuse dovrebbero essere riconosciute automaticamente. Se siete in grado di ascoltare gli esempi che vengono riprodotti durante la configurazione, premete solo OK ed ogni cosa sarà impostata per voi.

Se la scheda non venisse rilevata automaticamente, potreste essere omaggiati di un elenco di schede sonore e/o di loro proprietà tra cui scegliere. Dopo di ciò dovreste indicare la porta di I/O e le impostazioni di IRQ e DMA corrette. Informazioni su queste impostazioni possono essere trovate tra la documentazione della vostra scheda audio. Se siete in un sistema a doppio avvio con MS Windows, tali informazioni possono essere rinvenute anche nel Pannello di Controllo di Windows.

Se l'identificazione automatica della scheda sonora fallisce
Se la vostra scheda sonora non è supportata in partenza, avrete necessità di seguire altre tecniche: esse sono descritte nel Linux Sound HOWTO.

11.1.2. I driver e l'architettura

In genere esistono due tipi di architettura del suono: il più anziano Open Sound System o OSS, che funziona con tutti i sistemi simil-UNIX, ed il più recente Advanced Linux Sound Architecture o ALSA, che ha un migliore supporto per Linux, come indica il nome stesso. Alsa ha pure maggiori funzionalità e consente uno sviluppo più rapido dei driver. Qui ci concentreremo sul sistema ALSA.

Oggigiorno quasi tutti i correnti chipset audio vengono supportati: restano escluse solo alcune soluzioni professionali di alto livello e alcune schede sviluppate da produttori che si rifiutano di documentare le specifiche dei loro chipset. Una panoramica delle periferiche supportate può essere

rintracciata nel sito di ALSA in http://www.alsa-project.org/alsa-doc/index.php?vendor=All#matrix.

La configurazione dei sistemi dotati di ALSA si effettua con lo strumento **alsaconf**. Inoltre le distribuzioni di solito mettono a disposizione i propri strumenti per configurare la scheda sonora; tali strumenti potrebbero anche integrare il vecchio ed il nuovo modo di gestire le periferiche audio.

11.2. Riproduzione audio e video

11.2.1. Ascolto e copia dei CD

Il pacchetto *cdp* è contenuto nella maggioranza delle distribuzioni e fornisce **cdp** o **cdplay**, un riproduttore CD in modalità testuale. Normalmente i gestori di desktop includono uno strumento grafico (come **gnome-cd** sotto Gnome), che può essere avviato da un menu.

Siate certi di comprendere la differenza tra CD audio e CD dati. Non dovete montare un CD audio nel file system per ascoltarlo. Ciò perché i dati di quel tipo di CD non sono dati del file system di Linux; essi vengono letti ed inviati al canale d'uscita audio direttamente utilizzando un programma di riproduzione di CD. Se il vostro CD è un CD dati contenente file `.mp3`, prima avrete bisogno di montarlo nel file system e poi di usare uno dei programmi che tratteremo di seguito per riprodurre la musica. Come montare i CD nel file system è stato spiegato nella Sezione 7.5.5.

Lo strumento **cdparanoia** dal pacchetto omonimo legge l'audio direttamente come dati dal CD, senza conversioni in analogico, e scrive i dati in un file o incanalandolo in differenti formati, tra i quali il più popolare è, probabilmente, il `.wav`. Vari strumenti per la conversione ad altri formati, come `.mp3`, sono contenuti in molte distribuzioni o sono scaricabili come pacchetti separati. Il progetto GNU fornisce parecchi strumenti per riprodurre, estrarre e codificare CD, e gestori di database; guardate la sezione Free Software Directory, Audio per informazioni dettagliate.

La creazione di CD audio è semplificata, fra molti altri, dallo strumento **kaudiocreator** della suite KDE. E' dotato delle chiare informazioni del Centro Aiuti KDE.

La creazione di CD è trattata genericamente nella Sezione 9.2.2.

11.2.2. La riproduzione di file musicali

11.2.2.1. I file .mp3

Il popolare formato `.mp3` è ampiamente supportato nelle macchine Linux. La maggior parte delle distribuzioni comprende molti programmi che possono riprodurre questo tipo di file. Fra tante altre applicazioni, XMMS (che viene presentato nell'immagine qui di seguito) è una delle più largamente diffuse, in parte perché ha lo stesso aspetto e funzionamento dell'analogo strumento Windows.

Figura 11-1. Il riproduttore .mp3 XMMS

Parimenti molto diffusi per la riproduzione della musica sono Amarok, un'applicazione KDE, che sta guadagnando costantemente popolarità, e MPlayer, che può riprodurre anche film.

Restrizioni

Alcune distribuzioni non vi permettono di ascoltare degli MP3 senza modificare la configurazione: ciò è dovuto alle restrizioni della licenza sugli strumenti MP3. Potreste aver bisogno di installare software aggiuntivo per essere in grado di suonare la vostra musica.

In modalità testo potete usare il comando **mplayer**:

```
[tille@octarine ~]$ mplayer /opt/mp3/oriental/*.mp3
MPlayer 1.0pre7-RPM-3.4.2 (C) 2000-2005 MPlayer Team
CPU: Advanced Micro Devices Duron Spitfire (Family: 6, Stepping: 1)
Detected cache-line size is 64 bytes
CPUflags: MMX: 1 MMX2: 1 3DNow: 1 3DNow2: 1 SSE: 0 SSE2: 0
Playing /opt/oldopt/mp3/oriental/Mazika_Diana-Krozon_Super-Star_Ensani-Ma-
Bansak.mp3.
Cache fill: 1.17% (98304 bytes)      Audio file detected.
Clip info:
Title: Ensani-Ma-Bansak.mp3
Artist: Diana-Krozon
Album: Super-Star
Year:
Comment:
Genre: Unknown
==========================================================================
Opening audio decoder: [mp3lib] MPEG layer-2, layer-3
mpg123: Can't rewind stream by 450 bits!
AUDIO: 44100 Hz, 2 ch, s16le, 160.0 kbit/11.34% (ratio: 20000->176400)
Selected audio codec: [mp3] afm:mp3lib (mp3lib MPEG layer-2, layer-3)
==========================================================================
Checking audio filter chain for 44100Hz/2ch/s16le -> 44100Hz/2ch/s16le...
AF_pre: 44100Hz/2ch/s16le
AO: [oss] 44100Hz 2ch s16le (2 bps)
Building audio filter chain for 44100Hz/2ch/s16le -> 44100Hz/2ch/s16le...
Video: no video
Starting playback...
A: 227.8 (03:23:.1) 1.8% 12%
```

11.2.2.2. Altri formati

Discutere su tutti i possibili formati audio e sui modi per ascoltarli ci potrebbe condurre troppo lontano. Ecco una panoramica (incompleta) di altro software comune per la riproduzione e la manipolazione del suono:

- Ogg Vorbis: formato audio libero: guardate la directory audio GNU per gli strumenti - essi potrebbero essere inclusi anche nella vostra distribuzione. Il formato è stato sviluppato in quanto MP3 è soggetto a brevetti.
- Real audio e video: **realplay** della RealNetworks.
- SoX o Sound eXchange: in realtà un convertitore di suoni, si trova con il programma **play**. Riproduce `.wav`, `.ogg` e vari altri formati, compresi quelli binari grezzi.
- Playmidi: un riproduttore MIDI (guardate la directory GNU).
- AlsaPlayer: dal progetto Advanced Linux Sound Architecture (guardate il sito web di AlsaPlayer).
- **mplayer**: riproduce praticamente tutto, compresi i file mp3. Maggiori informazioni nel sito web MPlayerHQ.
- **hxplay**: supporta RealAudio e RealVideo, mp3, audio mp4, Flash, wav e altro ancora. Consultate HelixDNA (non tutti i componenti di questo software sono completamente liberi).
- **rhytmbox**: basato sulla struttura di GStreamer, può riprodurre tutto ciò che viene supportato da GStreamer, che si vanta di essere in grado di riprodurre tutto (date uno sguardo ai siti di **Rhytmbox** e di **GStreamer**).

Controllate la documentazione del vostro sistema e le pagine man per strumenti particolari e dettagliate spiegazioni su come usarli.

Non ho queste applicazioni nel mio sistema!
Molti degli strumenti e delle applicazioni trattati nelle sezioni precedenti sono software opzionali. E' perciò possibile che tali applicazioni non siano state installate di base, ma che possiate ritrovarle nella vostra distribuzione come pacchetti addizionali. Potrebbe anche succedere molto facilmente che l'applicazione che state cercando non ci sia per niente nella vostra distribuzione: In tal caso dovrete scaricarla dal sito web dell'applicazione.

11.2.2.3. Controllo del volume

aumix e **alsamixer** sono due comuni strumenti a caratteri per la regolazione dei comandi audio. Utilizzate i tasti freccia per variare le regolazioni. **alsamixer** ha un'interfaccia grafica quando viene avviato dal menu di Gnome o come **gnome-alsamixer** dalla linea di comando. Lo strumento **kmix** fa le stesse cose in KDE.

Tralasciando cosa scegliete di ascoltare come musica o altri suoni, ricordate che ci potrebbe essere altra gente a cui non interessa ascoltare voi o il vostro computer. Provate ad essere cortesi, specialmente nell'ambito dell'ufficio. Usate cuffie di qualità piuttosto quelle piccole da infilare nelle orecchie: ciò è meglio per i vostri timpani e causa meno distrazioni per i colleghi.

11.2.3. La registrazione

Anche stavolta sono disponibili vari strumenti che vi permettono di registrare voci e musica. Per la registrazione vocale potete usare **arecord** da linea di comando:

```
alexey@russia:~> arecord /var/tmp/myvoice.wav
```

```
Recording WAVE '/var/tmp/myvoice.wav' : Unsigned 8 bit, Rate 8000 Hz, Mono
Aborted by signal Interrups...
```

“Interrupt” o "Interruzione" significa che l'applicazione ha ricevuto un **Ctrl+C**. Ascoltate il suono campionato usando il semplice comando **play**.

Questa è una buona prova che potete eseguire prima di sperimentare applicazioni che richiedono un ingresso vocale come Voice over IP (VoIP). Tenete bene in mente che l'ingresso microfonico dovrebbe essere attivato. Se non sentite la vostra voce, controllate le impostazioni del suono. Spesso succede che il microfono è muto oppure che il volume è molto basso. Ciò può essere facilmente sistemato usando **alsamixer** o l'interfaccia grafica del sistema sonoro specifica della vostra distribuzione.

Sotto KDE potete avviare il programma di utilità **krec**; Gnome fornisce **gnome-sound-recorder** (il registratore gnome del suono).

11.3. Riproduzione video, guardare flussi e televisione

Sono disponibili svariati riproduttori:

- **xine**: un riproduttore video libero
- **ogle**: riproduttore di DVD.
- **okle**: versione KDE di **ogle**.
- **mplayer**: Movie Player per Linux.
- **totem**: riproduce sia file audio che video, CD audio, VCD e DVD.
- **realplay**: da RealNetworks.
- **hxplay**: una alternativa a Real, consultate HelixDNA.
- **kaffeine**: riproduttore file multimediali di KDE3.

Assai probabilmente troverete uno di questi nei vostri menu grafici.

Tenete in mente che tutti i codec necessari per la visione dei diversi tipi di video potrebbero non essere stati installati inizialmente nel vostro sistema. Potreste dover percorrere una lunga strada per scaricare i w32codec e libdvdcss.

LPD ha rilasciato un documento che è molto appropriato per questa sezione: si intitola DVD Playback HOWTO e descrive i vari strumenti disponibili per la visione di DVD in un sistema dotato di lettore DVD. Si tratta di una buona appendice a DVD HOWTO che spiega l'installazione del lettore.

Per guardare la TV c'è da scegliere tra questi strumenti, oltre a molti altri per la visione e la cattura di TV, video ed altri flussi di dati:

- **tvtime**: grande programma con gestione delle stazioni, interazione con il televideo, modalità film e molto altro ancora.
- **zapping**: visualizzatore TV specifico di Gnome.
- **xawtv**: visualizzatore TV X11.

11.4. Telefonia Internet

11.4.1. Che cos'è?

La telefonia Internet, o più comunemente, Voice over IP (VoIP) o telefonia digitale, consente alle parti di scambiare flussi di dati vocali in rete. La grossa differenza è che i flussi di dati viaggiano in una rete ad uso generale, Internet, contrariamente alla telefonia convenzionale, che utilizza reti dedicate di linee di trasmissione vocale. Comunque le due reti possono essere connesse a determinate condizioni, ma per ora non si tratta di uno standard. In altri termini: è estremamente probabile che non sarete in grado di chiamare persone che stanno usando un telefono convenzionale. Se nonostante tutto ciò è possibile, è probabile che dobbiate pagare un abbonamento.

Sebbene attualmente siano disponibili svariate applicazioni da scaricare gratuitamente, sia libere che proprietarie, esistono alcuni grossi inconvenienti nella telefonia su Internet: tra quelli maggiormente degni di nota, il sistema è inaffidabile, può essere lento o ci può essere molto rumore nella connessione e così certamente non può venire usato per rimpiazzare la telefonia convenzionale (pensate alle chiamate d'emergenza). Quantunque alcuni provider prendano le loro precauzioni, non esiste alcuna garanzia che voi possiate raggiungere il soggetto che intendete chiamare.

Attualmente molte applicazioni non usano la crittografia, per cui state attenti che è potenzialmente facile per qualcuno ascoltare di nascosto le vostre conversazioni. Se la sicurezza è una vostra preoccupazione, leggete la documentazione allegata al vostro cliente VoIP. In aggiunta, se state utilizzando un firewall, questo dovrebbe essere configurato per permettere connessioni da qualsiasi provenienza: perciò l'uso del VoIP implica l'assunzione di rischi a livello della sicurezza del sito.

11.4.2. Cosa vi serve?

11.4.2.1. Il lato server

Innanzitutto, avete bisogno di un provider che vi offra il servizio. Questo servizio potrebbe integrare la telefonia tradizionale e potrebbe essere gratuito o meno. Fra gli altri ci sono SIPphone, Vonage, Lingo, AOL TotalTalk e molti altri provider accessibili localmente che offrono il cosiddetto "servizio telefonico completo" [full phone service]. Il solo servizio di telefonia Internet è offerto da Skype, SIP Broker, Google e molti altri.

Se voleste realizzare un server da voi stessi, dovreste dare un occhio ad Asterisk.

11.4.2.2. Il lato cliente

Sul lato cliente le applicazioni che potete utilizzare dipendono dalla configurazione della vostra rete. Se avete una connessione Internet diretta, non dovrebbero esserci problemi, visto che sapete a quale server vi potete connettere e che normalmente avete anche un nome utente e una password per autenticarvi al servizio.

Comunque, se vi trovate dietro ad un firewall che fa la "traduzione degli indirizzi di rete" (NAT o Network Address Translation), alcuni servizi potrebbero non funzionare dal momento che essi vedrebbero solo l'indirizzo IP del firewall e non quello del computer, che potrebbe essere non instradabile su Internet (per esempio quando siete in una rete aziendale e il vostro indirizzo IP inizia con 10., 192.168. o un altro prefisso di sottorete non instradabile). Ciò dipende dal protocollo utilizzato dall'applicazione.

Anche la larghezza di banda disponibile potrebbe essere un fattore bloccante: alcune applicazioni sono ottimizzate per un ridotto consumo di larghezza di banda, mentre altre potrebbero richiedere connessioni ad alta larghezza di banda. Ciò dipende dal codec utilizzato dall'applicazione.

Fra le più comuni applicazioni ci sono il cliente Skype, che ha un'interfaccia che ricorda i programmi di messaggeria istantanea, e X-Lite, la versione gratuita del telefono software XTen, che assomiglia ad un telefono cellulare. Comunque, mentre questi programmi sono disponibile per lo scaricamento gratuito e sono molto popolari, essi non sono liberi come la parola libera: adottano protocolli proprietari e/o sono forniti esclusivamente in pacchetti binari, non in formato sorgente.

Clienti VoIP gratuiti e *aperti* sono per esempio Gizmo, Linphone, GnomeMeeting e Kphone.

Hardware del cliente
Sebbene il vostro computer, specialmente se è un PC portatile, possa avere un microfono integrato, la resa sarebbe molto migliore collegando delle cuffie con microfono. Se si tratta di scegliere, optate per delle cuffie USB in quanto funzionano indipendentemente dall'hardware audio esistente. Usate **alsamixer** per configurare i livelli d'ingresso e d'uscita del suono a vostro piacimento.

Le applicazioni VoIP sono assolutamente un mercato in espansione. Dei volontari tentano di documentare lo stato attuale su http://www.voip-info.org/.

11.5. Sommario

La piattaforma GNU/Linux è pienamente abilitata alla multimedialità. Viene supportata una ampia varietà di periferiche come le schede sonore, le schede TV, le cuffie microfoniche, i lettori di CD e DVD. L'elenco delle applicazioni è semplicemente infinito: è per questo motivo che abbiamo dovuto ridurre il seguente elenco dei nuovi comandi e limitare noi stessi ai comandi audio generali.

Tabella 11-1. Nuovi comandi nel capitolo 11: Audio

Comando	Significato
alsaconf	Configura il sistema sonoro ALSA.
alsamixer	Regola i livelli di uscita del driver ALSA.
arecord	Registra un campionamento di suono.
aumix	Strumento per il mixer audio.

Comando	Significato
cdp	Riproduce un CD audio.
cdparanoia	Estrae le tracce da un CD audio.
cdplay	Riproduce un CD audio.
gnome-alsamixer	Programma di interfaccia a Gnome ALSA.
gnome-cd	Programma di interfaccia Gnome per l'ascolto di CD audio
gnome-sound-recorder	Programma di interfaccia Gnome per la registrazione di campioni sonori.
kaudiocreator	Programma di interfaccia KDE per la creazione di CD audio.
kmix	Programma di interfaccia KDE per la regolazione del suono.
krec	Programma di interfaccia KDE per la registrazione di campioni sonori.
mplayer	Riproduttore multimediale.
play	Strumento a linea di comando per l'ascolto di campioni sonori.

11.6. Esercizi

1) Dal menu di Gnome o di KDE aprite il pannello della configurazione sonora. Accertatevi che gli altoparlanti o le cuffie siano connesse al vostro sistema e trovate un livello di emissione che sia per voi confortevole. Assicuratevi, quando il vostro sistema è ALSA compatibile, di usare il pannello giusto.
2) Se avete un microfono, provate a registrare un campione della vostra voce. Accertatevi che il volume in ingresso non sia troppo alto, dal momento che ciò comporterebbe toni con picchi elevati mentre comunicate con altri oppure la trasmissione di rumori di sottofondo all'altra parte. Nella linea di comando potreste anche usare **arecord** o **aplay** per registrare ed ascoltare il suono.
3) Localizzate i file sonori nel vostro sistema e provate ad ascoltarli.
4) Inserite un CD audio e provate ad ascoltarlo.
5) Trovate un compagno di chat e configurate un programma VoIP (dovreste prima installarne uno).
6) Potete ascoltare le radio Internet?
7) Se avete un lettore DVD e un film su disco DVD, provate a riprodurlo.

Appendice A. Dove andare da qui?

Questo documento offre una panoramica di libri e siti utili.

A.1. Libri utili

A.1.1. Linux in generale

- "Linux in a Nutshell" di Ellen Siever, Jessica P. Hackman, Stephen Spainhour, Stephen Figgins, O'Reilly UK, ISBN 0596000251
- "Running Linux" di Matt Welsh, Matthias Kalle Dalheimer, Lar Kafman, O'Reilly UK, ISBN 156592469X
- "Linux Unleashed" di Tim Parker, Bill Bal, David Pitts, Sams, ISBN 0672316889
- "When You Can't Find Your System Administrator" di Linda Mui, O'Reilly UK, ISBN 1565921046
- Quando acquistate veramente una distribuzione, essa conterrà un discreto manuale per l'utente.

A.1.2. Editor

- "Learning the Vi Editor" di Linda Lamb e Arnold Robbins, O"Reilly UK, ISBN 1565924266
- "GNU Emacs Manual" di Richard M. Stallman, iUniverse.Com Inc., ISBN 0595100333
- "Learning GNU Emacs" di Debra Cameron, Bill Rosenblatt ed Eric Raymond, O"Reilly UK, ISBN 1565921526
- "Perl Cookbook" di Tom Christiansen e Nathan Torkington, O'Reilly UK, ISBN 156592433

A.1.3. Shell

- "Unix Shell Programming" di Stephen G. Kochan e Patrick H. Wood, Sams Publishing, ISBN 067248448X
- "Learning the Bash Shell" di Cameron Newham e Bill Rosenblatt, O'Reilly UK, ISBN 1565923472
- "Linux and Unix Shell Programming" di David Tansley, Addison Wesley Pubblishing Company, ISBN 0201674726
- "Unix C Shell Field Guide" d Gail e Paul Anderson, Prentice Hall, ISBN 013937468X

A.1.4. X Window

- "Gnome User's Guide" della Comunità Gnome, iUniverse.Com Inc., ISBN 0595132251
- "KDE Bible" di Dave Nash, Hungry Minds Inc., ISBN 0764546929
- "The Concise Guide to XFree86 for Linux" di Aron Hsiao, Que, ISBN 0789721821
- "The New XFree86" di Bill Ball, Prima Publishing, ISBN 0761531521
- "Beginning GTK+ and Gnome" di Peter Wright, Wrox Press, ISBN 1861003811
- "KDE 2.0 Development" di David Sweet e Matthias Ettrich, Sams Publishing, ISBN

0672318911

- "GTK+/Gnome Application Development" di Havoc Pennington, New Riders Publishing, ISBN 0735700788

A.1.5. Reti

- "TCP/IP Illustrated, Volume I: The Protocols" di W. Richard Stevens, Addison-Wesley Professional Computing Serie, ISBN 0-2001-63346-9
- "DNS and BIND" di Paul Albitz, Cricket Liu, Mike Loukides e Deborah Russell, O'Reilly & Associates, ISBN 0696001584
- "The Concise Guide to DNS and Bind" di Nicolai Langfeldt, Que, ISBN 0789722739
- "Implementing LDAP" di Mark Wilcox, Wrox Press, ISBN 1861002211
- "Understanding and deploying LDAP directory services" di Tim Howes and co., Sams, ISBN 0672323168
- "Sendmail" di Brian Costales e Eric Allman, O'Reilly UK, ISBN 1565922220
- "Removing the Spam: Email Processing and Filtering" di Geoff Mulligan, Addison Wesley Publishing Company, ISBN 0201379570
- "Managing IMAP" di Dianna & Kevin Mullet, O'Reilly UK, ISBN 059600012X

A.2. Siti utili

A.2.1. Informazioni generali

- Linux Documentation Project: tutti i documenti, le pagine man, gli HOWTO, le FAQ
- LinuxQuestion.org: forum, scaricamenti, documenti e molto ancora
- Google for Linux: il motore di ricerca specializzato
- Google Groups: un archivio di tutti i messaggi dei newsgroup, compresa la gerarchia comp.os.linux
- Slashdot: notizie giornaliere
- http://www.oreilly.com: libri sul sistema Linux e l'amministrazione di rete, Perl, Java, ...
- POSIX: lo standard
- Linux HQ: gestisce un database completo dei sorgenti, delle pezze e della documentazione di varie versioni del kernel di Linux.

A.2.2. Riferimenti a specifiche architetture

- AlphaLinux: Linux su architettura Alpha (ad es. Workstation Digital)
- Linux-MIPS: Linux su MIPS (ad es. SGI Indy)
- Linux on the Road: istruzioni specifiche per installare e far girare Linux su portatili, palmari, telefoni cellulari e così via. File di configurazione per svariati modelli
- MkLinux: Linux su Apple

A.2.3 Distribuzioni

- Progetto Fedora: OS sostenuto dalla comunità sponsorizzata da RedHat
- Mandriva

- Ubuntu
- Debian
- TurboLinux
- Slackware
- SuSE
- LinuxIso.org: immagini di CD per tutte le distribuzioni
- Knoppix: distribuzione che gira su CD; non dovete installare nulla da quest'ultimo
- DistroWatch.com: scovate un Linux che va d'accordo con il vostro stile
- ...

A.2.4. Software

- Freshmeat: software nuovo, archivi di software
- OpenSSH: sito di Secure SHell
- OpenOffice: suite da ufficio compatibile MS
- KDE: sito di K Desktop
- GNU: GNU e software GNU
- Gnome: il sito ufficiale Gnome
- RPM Find: tutti i pacchetti RPM
- Samba: servizi MS Windows di file e stampa
- Home of the OpenLDAP Project: server/clienti/utilità OpenLDAP, FAQ e altra documentazione
- Sendmail Homepage: una approfondita trattazione tecnica delle caratteristiche di Sendmail, comprendente esempi di configurazione
- Netfilter: contiene una miscellanea di informazioni su iptables: HOWTO, FAQ, guide, ...
- Sito ufficiale GIMP: tutte le informazioni su GNU Image Manipulation Program [programma GNU di manipolazione delle immagini]
- SourceForge.net: sito di sviluppo del software a sorgente aperto
- homepage di vIm

Appendice B. Comandi DOS contro Linux

In questa appendice confrontiamo i comandi DOS con i loro equivalenti Linux.

Quale ulteriore mezzo di orientamento per i nuovi utenti con esperienze di Windows, la tabella seguente elenca dei comandi MS-DOS con i loro corrispondenti in Linux. Tenete in mente che solitamente i comandi Linux hanno un certo numero di opzioni. Leggete le pagine Info o man sul comando per scoprire di più.

Tabella B-1. Panoramica dei comandi DOS/Linux

Comandi DOS	Comandi Linux
<comando> /?	**man <comando>** oppure **comando** `--help`
cd	**cd**
chdir	**pwd**
cls	**clear**
copy	**cp**
date	**date**
del	**rm**
dir	**ls**
echo	**echo**
edit	**vim** (o altro editor)
exit	**exit**
fc	**diff**
find	**grep**
format	**mke2fs** oppure **mformat**
mem	**free**
mkdir	**mkdir**
more	**more** oppure **less**
move	**mv**
ren	**mv**
time	**date**

Appendice C. Caratteristiche della shell

Questo documento offre una panoramica delle comuni caratteristiche della shell (le stesse in ogni tipo di shell) e quelle differenti (specifiche di ciascuna shell).

C.1. Caratteristiche comuni

Le seguenti caratteristiche sono standard in ogni shell. Notate che i comandi stop, suspend, jobs, bg e fg sono disponibili solo nei sistemi che supportano il controllo dei processi.

Tabella C-1. Caratteristiche comuni delle shell

Comando	Significato
>	Redirige l'output
>>	Aggiunge al file
<	Redirige l'input
<<	Documento “Qui” (redirige l'input)
|	Incanalamento dell'output
&	Esegue il processo sullo sfondo
;	Separa comandi sulla stessa linea
*	Usa qualsiasi carattere nel nome del file
?	Sostituisce un singolo carattere nel nome del file
[ ]	Usa qualsiasi carattere racchiuso
( )	Esegue in una sottoshell
` `	Sostituisce l'output del comando racchiuso
“ ”	Apici parziali (consente l'espansione di variabili e comandi)
' '	Apici totali (nessuna espansione)
\	Impedisce l'interpretazione del carattere successivo
$var	Usa il valore per la variabile
$$	ID del processo
$0	Nome del comando
$n	ennesimo argomento (n da 0 a 9)
$*	Tutti gli argomenti come una semplice parola
#	Inizio di un commento

Comando	Significato
bg	Esecuzione sullo sfondo
break	Interruzione da comandi ciclici
cd	Cambia directory
continue	Riavvia un ciclo di programma
echo	Mostra l'output
eval	Valuta degli argomenti
exec	Esegue una nuova shell
fg	Esecuzione in primo piano
jobs	Mostra i processi attivi
kill	Termina i processi in funzione
newgrp	Cambia ad un nuovo gruppo
shift	Parametri posizionali di spostamento
stop	Sospende un processo sullo sfondo
suspend	Sospende un processo in primo piano
time	Cronometra un programma
umask	Imposta o elenca i permessi dei file
unset	Cancella le definizioni delle variabili o delle funzioni
wait	Attende che finisca un processo sullo sfondo

C.2. Caratteristiche diverse

La tabella seguente mostra le principali differenze tra la shell standard (**sh**), Bourne Again SHell (**bash**), Korn shell (**ksh**) e la C shell (**csh**).

Compatibilità delle shell
Dal momento che Bourne Again SHell è un superinsieme di **sh**, tutti i comandi **sh** funzioneranno anche con **bash** - ma non viceversa. **bash** possiede molte altre caratteristiche proprie e, come dimostra la seguente tabella, molte di queste sono state incorporate prendendole da altre shell.

Poiché la Turbo C shell è un superinsieme di **csh**, tutti i comandi **csh** funzioneranno con **tcsh**, ma non il contrario.

Tabella C-2. Caratteristiche differenti delle shell

sh	bash	ksh	csh	Significato/Azione
$	$	$	%	Prompt base dell'utente
	>|	>|	>!	Forza la redirezione
> file 2>&1	&> file oppure > file 2>&1	> file 2>&1	>& file	Redirige stdout e stderror a `file`
	{ }		{ }	Espande gli elementi in elenco
\`comando\`	**\`comando\`** oppure **$(comando)**	**$(comando)**	**\`comando\`**	Sostituisce l'output del **comando** racchiuso
`$HOME`	`$HOME`	`$HOME`	`$home`	Directory personale (home)
	~	~	~	Simbolo della directory personale
	~+, `~-`, **dirs**	~+, `~-`	`=-`, `=N`	Stack della directory di accesso
`var`=valore	**VAR=valore**	**`var`=valore**	**set `var`=valore**	Assegnazione di variabile
export `var`	**export VAR=value**	**export `var`=val**	**setenv `var` *`val`***	Imposta la variabile d'ambiente
	`${nnnn}`	`${nn}`		Possono essere referenziati più di 9 argomenti
`"$@"`	`"$@"`	`"$@"`		Tutti gli argomenti come parole separate
`$#`	`$#`	`$#`	`$#argv`	Numero di argomenti
`$?`	`$?`	`$?`	`$status`	Stato di uscita del più recente comando eseguito
`$!`	`$!`	`$!`		PID del processo messo sullo sfondo più di recente
`$-`	`$-`	`$-`		Opzioni correnti
`. file`	**source `file`** oppure **`. file`**	**`. file`**	**source `file`**	Legge i comandi in file
	alias x='y'	**alias x=y**	**alias x y**	Il nome **x** sta al posto del comando **y**
case	**case**	**case**	**switch** o **case**	Alternative della scelta
done	**done**	**done**	**end**	Termina un comando di ciclo
esac	**esac**	**esac**	**endsw**	Fine di **switch** o **case**
exit *n*	**exit *n***	**exit *n***	**exit *`(expr)`***	Uscita con uno stato
for/do	**for/do**	**for/do**	**foreach**	Cicla con le variabili

sh	bash	ksh	csh	Significato/Azione
	set -f, set –o nullglob|dotglob|nocaseglob|noglob		noglob	Ignora i caratteri di sostituzione per la generazione del nome del file
hash	hash	alias -t	hashstat	Mostra i comandi sottoposti a hash (alias tracciati)
hash *cmds*	hash *cmds*	hash -t *cmds*	rehash	Ricorda la posizione dei comandi
hash -r	hash -r		unhash	Dimentica la posizione dei comandi
	history	history	history	Elenca i comandi precedenti
	FrecciaSu+Enter oppure **!!**	r	!!	Riesegue il comando precedente
	!*str*	r *str*	!*str*	Riesegue l'ultimo comando che inizia con "str"
	!*cmd*:s/*x*/*y*/	r *x*=*y* *cmd*	!*cmd*:s/*x*/*y*/	Sostituisce "x" con y" nel più recente comando che inizia con "cmd" e poi lo esegue
if [$i –eq 5]	if [$i –eq 5]	if ((i==5))	if ((i==5))	Prova della condizione di esempio
fi	fi	fi	endif	Termina l'istruzione **if**
ulimit	ulimit	ulimit	limit	Imposta i limiti delle risorse
pwd	pwd	pwd	dirs	Stampa la directory in uso
read	read	read	$<	Legge dal terminale
trap *2*	trap *2*	trap *2*	onintr	Ignora le interruzioni
	unalias	unalias	unalias	Rimuove gli alias
until	until	until		Inizia un ciclo **until**
while/do	while/do	while/do	while	Inizia un ciclo **while**

La Bourne Again SHell possiede molte caratteristiche qui non riportate. Questa tabella è solo per darvi un'idea di come tale shell incorpori tutte le trovate utili delle altre shell: non ci sono spazi vuoti nella colonna di **bash**. Ulteriori informazioni sulle caratteristiche riscontrabili solamente in Bash si possono rintracciare nelle pagine info di Bash, nella sezione "Bash Features".

Maggiori informazioni:

dovreste leggere almeno un manuale, quello della vostra shell. La scelta preferibile potrebbe essere **info bash**, essendo **bash** la shell GNU e la più semplice per i principianti. Stampatelo, portatevelo a casa e studiatelo ogni volta che avete 5 minuti.

Guardate l'Appendice B se avete difficoltà ad assimilare i comandi della shell.

Appendice D. GNU Free Documentation License

D.1. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

D.2. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a

section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

D.3. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying

this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

D.4. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

D.5. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

GNU FDL Modification Conditions

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

D.6. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

D.7. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

D.8. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

D.9. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

D.10. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

D.11. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

D.12. ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> **Sample Invariant Sections list**
>
> Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

> **Sample Invariant Sections list**
>
> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Glossario

Questa sezione contiene una panoramica in ordine alfabetico dei comandi trattati nel presente documento.

A

a2ps
Formatta i file per la stampa con una stampante PostScript (v. Sezione 8.1.2.).

acroread
Visualizzatore PDF (v. Sezione 8.1.2.2.)

adduser
Crea un nuovo utente o aggiorna le informazioni base del nuovo utente.

alias
Crea un alias di shell per un comando.

alsaconf
Configura la scheda sonora usando il driver ALSA (v. Sezione 11.1.2.).

alsamixer
Regola l'uscita del dispositivo sonoro ALSA (v. Sezione 11.2.2.3.).

anacron
Esegue periodicamente dei comandi, supponendo che la macchina non funzioni continuamente.

apropos
Cerca stringhe nel database di whatis (v. Sezione 2.3.3.2.).

apt-get
Il programma di utilità per la gestione dei pacchetti di APT (v. Sezione 7.5.3.2.).

arecord
Registra un campione sonoro (v. Sezione 11.2.3.)

aspell
Correttore delle parole

at, **atq**, **atrm**
Accoda, esamina o cancella processi per un'esecuzione ritardata (v. Sezione 4.1.2.2.).

aumix
Regola il mixer audio (v. Sezione 11.2.2.3.).

(g)awk
Linguaggio di scansione ed elaborazione di schemi.

B

bash
Bourne Again SHell (v. Sezione 3.2.3.2. e Sezione 7.2.5.).

batch
Accoda, esamina o cancella processi ad esecuzione ritardata (v. Sezione 4.1.2.2.).

bg
Esegue un processo in sottofondo (v. Sezione 4.1.2.1.).

bitmap
Utilità di elaborazione e conversione di bitmap per il sistema X Window.

bzip2
Un compressore di file ad ordinamento di blocchi (v. Sezione 9.1.1.3.).

C

cardctl
Gestisce le schede PCMCIA (v. Sezione 10.2.3.3.).

cat
Concatena file e li stampa nello standard output (v. Sezione 2.2. e Sezione 3.2.4.).

cd
Cambia directory (v. Sezione 2.2.).

cdp/cdplay
Un programma interattivo in modalità testo per controlla e riprodurre CD Rom audio sotto Linux (v. Sezione 11.2.1.).

cdparanoia
Un'utilità di lettura di CD audio che comprende caratteristiche aggiuntive di verificazione dei dati (v. Sezione 11.2.1.).

cdrecord
Registra un CD-R (v. Sezione 9.2.2.).

chattr
Cambia gli attributi dei file.

chgrp
Cambia la proprietà del gruppo (v. Sezione 3.4.2.3.).

chkconfig
Aggiorna o ricerca informazioni di livello d'esecuzione per i servizi di sistema (v. Sezione 4.2.5.1.).

chmod
Cambia i permessi di accesso al file (v. Sezione 3.4.1., Sezione 3.4.2.1. e Sezione 3.4.2.4.).

chown
Cambia il proprietario e il gruppo di un file (v. Sezione 3.4.2.3.).

compress
Comprime file.

cp
Copia file e directory (v. Sezione 3.2.2.).

crontab
Gestisce i file crontab (v. Sezione 4.4.4.).

csh
Apre una C shell (v. Sezione 3.2.3.2.).

cut
Rimuove sezioni da ogni linea di file (v. Sezione 7.2.5.2.).

D

date
Stampa o imposta ora e data del sistema.

dd
Converte e copia un file (disk dump) (v. Sezione 9.2.1.2.).

df
Riporta l'utilizzo del disco da parte del file system (v. Sezione 3.1.2.3.).

dhcpcd
Demone cliente DHCP (v. Sezione 10.3.8.).

diff
Scopre le differenze tra due file.

dig
Invia pacchetti di ricerca del nome di dominio a server dei nomi (v. Sezione 10.2.6.1.).

dmesg
Stampa o controlla il kernel ring buffer.

du
Stima l'uso di spazio da parte del file.

dump
Esegue un salvataggio di sicurezza del file system (v. Sezione 9.2.5.).

E

echo
Visualizza una linea di testo (v. Sezione 3.2.1.).

ediff
Traduttore da diff all'inglese.

egrep
Extended grep.

eject
Smonta ed espelle supporti removibili (v. Sezione 7.5.5.2.).

emacs
Avvia l'editor Emacs (v. Sezione 6.1.2.2.).

exec
Invoca sottoprocessi (v. Sezione 4.1.5.1.).

exit
Esce dalla shell corrente (v. Sezione 2.2.).

export
Aggiunge funzioni all'ambiente di shell (v. Sezione 3.2.1., Sezione 7.2.1.2. e Sezione 7.2.4.2.).

F

fax2ps
Converte un facsimile TIFF in PostScript (v, Sezione 8.1.2.).

fdformat
Formatta dischi floppy (v. Sezione 9.2.1.1).

fdisk
Gestore Linux della tabella delle partizioni (v. Sezione 3.1.2.2.).

fetchmail
Preleva la posta da un server POP, IMAP, abilitato ETRN o ODMR (v. Sezione 10.3.2.3.).

fg
Porta in primo piano un processo (v. Sezione 4.1.2.1.).

file
Determina il tipo di file (v. Sezione 3.3.1.2.).

find
Trova file (v. Sezione 3.3.3.3.).

firefox
Navigatore di rete (v. Sezione 10.3.3.2.).

fork
Crea un nuovo processo (v. Sezione 4.1.5.1.).

formail
(Ri)organizzatore della posta (v. Sezione 10.3.2.3.).

fortune
Stampa una probabilmente interessante frase a caso.

ftp
Servizi di trasferimento file (insicuri a meno che venga utilizzato l'account anonimo!)(v. Sezione 10.3.4.2.).

G

galeon
Navigatore grafico della Rete.

gdm
Gnome Display Manager (v. Sezione 4.2.4.).

gedit
Editor GUI (v. Sezione 6.3.3.3.).

(min/a)getty
Controlla le periferiche della console.

gimp
Programma di elaborazione di immagini.

gpg
Codifica, controlla e decodifica i file (v. Sezione 9.4.1.2.).

grep
Stampa le linee che coincidono con uno schema (v. Sezione 3.3.3.4 e Sezione 5.3.1.).

groff
Emulate il comando nroff con groff (v. Sezione 8.1.2.).

grub
Shell di grub (v. Sezione 4.2.3. e Sezione 7.5.4.).

gv
Un visualizzatore PostScript e PDF (v. Sezione 8.1.2.2.).

gvim
Versione grafica dell'editor vIm (v. Sezione 6.3.3.3.).

gzip
Comprime ed espande file (v. Sezione 9.1.1.3.).

H

halt
Ferma il sistema (v. Sezione 4.2.6.).

head
Restituisce la prima parte dei file (v. Sezione 3.3.4.3.).

help
Mostra gli aiuti in un comando integrato nella shell.

host
Utilità di DNS lookup (v. Sezione 10.2.6.1).

httpd
Server Apache del protocollo di trasferimento ipertestuale (http) (v. Sezione 10.3.3.1.).

I

id
Stampa i veri ed effettivi UID e GID (v. Sezione 3.4.1.).

ifconfig
Configura l'interfaccia di rete o ne mostra la configurazione (v. Sezione 10.1.2.3.).

info
Legge i documenti Info (v. Sezione 2.3.3.1.).

init
Inizializzazione del controllo dei processi (v. Sezione 4.1.5.1., Sezione 4.2.4. e Sezione 4.2.5.).

innsserv
Gestisce gli script di init (v. Sezione 4.2.5.1.).

iostat
Mostra le statistiche di I/O (v. Sezione 4.3.5.4.).

ip
Mostra/cambia lo stato dell'interfaccia di rete (v. Sezione 10.2.3.1.).

ipchains
Amministrazione del firewall IP (v. Sezione 10.5.4.2.).

iptables
Amministrazione del filtro dei pacchetti IP (v. Sezione 10.5.4.2.).

J

jar
Strumento di archiviazione Java (v. Sezione 9.1.1.4.).

jobs

Elenca i processi in sottofondo.

K

kdm
Gestore desktop di KDE (v. Sezione 4.2.4.).

kedit
Editor grafico di KDE (v. Sezione 6.3.3.3.).

kill(all)
Termina i processi (v. Sezione 4.1.2.1.).

konqueror
Gestore di file, navigatore (aiuti) (v. Sezione 3.3.2.1.).

ksh
Apre una shell Korn (v. Sezione 3.2.3.2.).

kwrite
Editor grafico di KDE (v. Sezione 6.3.3.3.).

L

less
more con maggiori caratteristiche (v. Sezione 3.3.4.2.).

lilo
LInux boot LOader (v. Sezione 4.2.).

links
Navigatore WWW in modalità testo (v. Sezione 10.2.3.2.).

ln
Crea collegamenti tra file (v. Sezione 3.3.5.).

loadkeys
Carica le tabelle di traduzione della tastiera (v. Sezione 7.4.1.).

locate
Trova file (v. Sezione 3.3.3.3. e Sezione 4.4.4.).

logout
Chiude la shell corrente (v. Sezione 2.1.3.).

lp
Invia richieste al servizio di stampa LP (v. Sezione 8.1.).

lpc
Programma di controllo della stampante di linea (v. Sezione 8.1.).

lpq
Programma di verifica delle code di attesa di stampa (v. Sezione 8.1.).

lpr
Stampa offline (v. Sezione 8.1.).

lprm
Rimuove le richieste di stampa (v. Sezione 8.1.).

ls
Lista il contenuto della directory (v. Sezione 2.2., Sezione 3.1.1.2. e Sezione 3.3.1.1.).

lynx
Navigatore WWW in modalità testo (v. Sezione 10.2.3.2.).

M

mail
Invia e riceve posta (v. Sezione 10.3.2.3.).

man
Legge le pagine man (v. Sezione 2.3.2.).

mc
Midnight Commander, gestore di file (v. Sezione 3.3.2.1.).

mcopy
Copia file MSDOS verso/da Unix.

mdir
Mostra una directory MSDOS.

memusage
Mostra l'uso della memoria (v. Sezione 4.3.5.3.).

memusagestat
Mostra le statistiche sull'uso della memoria (v. Sezione 4.3.5.3.).

mesg
Controlla gli accessi in scrittura al vostro terminale (v. Sezione 4.1.6.).

mformat
Aggiunge un file system MSDOS ad un disco floppy formattato a basso livello (v. Sezione 9.2.1.1.).

mkbootdisk
Crea un floppy di avvio autonomo per la partenza del sistema.

mkdir
Crea una directory (v. Sezione 3.3.2.).

mkisofs
Crea un file system ibrido ISO9660 (v. Sezione 9.2.2.).

more
Filtra per mostrare del testo uno schermo alla volta (v. Sezione 3.3.4.2.).

mount
Monta un file system o mostra le informazioni relative ai file montati (v. Sezione 7.5.5.1.).

mozilla
Navigatore web (v. Sezione 10.2.3.2.).

mplayer
Riproduttore/codificatore di film per Linux (v. Sezione 11.2.2. e Sezione 11.3.).

mt
Controlla le operazioni relative all'unità a nastro magnetico.

mtr
Strumento di diagnosi per la rete.

mv
Rinomina i file (v. Sezione 3.3.2.).

N

named
Server dei nomi di dominio Internet (v. Sezione 10.3.7.).

nautilus
Gestore di file (v. Sezione 3.3.2.1.).

ncftp
Programma navigatore per servizi ftp (non sicuro!) (v. Sezione 10.3.4.2.).

netstat
Stampa le connessioni di rete, le tabelle di instradamento, le connessioni mascherate e le appartenenze al multicast (v. Sezione 10.2.5. e Sezione 10.5.2.).

newgrp
Iscrive in un un altro gruppo (v. Sezione 3.4.2.2.).

nfsstat
Stampa le statistiche sui file system in rete.

nice
Avvia un programma con la priorità di esecuzione modificata (v. Sezione 4.3.5.1.).

nmap
Strumento di esplorazione della rete e scanner della sicurezza.

ntpd
Network Time Protocol Daemon (v. Sezione 7.4.3.).

ntpdate
Imposta data e ora tramite un server NTP (v. Sezione 7.4.3.).

ntsysv
Semplice interfaccia per configurare i livelli di esecuzione (v. Sezione 4.2.5.1.).

O

ogle
Lettore di DVD con il supporto dei menu DVD (v. Sezione 11.3.).

P

passwd
Cambia la password (v. Sezione 2.2. e Sezione 4.1.6.).

pccardctl

Gestisce le schede PCMCIA (v. Sezione 10.2.3.3.).

pdf2ps
Convertitore GhostScript da PDF a PostScript (v. Sezione 8.1.2.).

perl
Practical Extraction and Report Language.

pg
Output testuale di pagina (v. Sezione 3.3.4.2.).

pgrep
Ricerca di processi in base al nome od altri attributi (v. Sezione 4.1.4.).

ping
Invia una richiesta di eco ad un host (v. Sezione 10.2.6.2.).

play
Riproduce un campione sonoro (v. Sezione 11.2.3.).

pr
Converte file di testo per la stampa.

printenv
Stampa tutto o parte dell'ambiente (v. Sezione 7.2.1.).

procmail
Elaboratore di posta autonomo (v. Sezione 10.3.2.3.).

ps
Restituisce lo stato dei processi (v. Sezione 4.1.4. e Sezione 4.3.5.4.).

pstree
Mostra un albero dei processi (v. Sezione 4.1.4.).

pwd
Stampa la Present Working Directory, la directory di lavoro corrente (v. Sezione 2.2.).

Q

quota
Mostra l'uso dei dischi ed i limiti (v. Sezione 3.2.3.3.).

R

rcp
Copia in remoto (non sicuro!).

rdesktop
Cliente di protocollo Remote Desktop (v. Sezione 10.4.6.).

recode
Converte i file in un altro insieme di caratteri (v. Sezione 7.4.4.).

renice
Altera la priorità di esecuzione dei processi (v. Sezione 4.3.5.1.).

rlogin
Login remoto (telnet, non sicuro!) (v. Sezione 10.4.2. e Sezione 10.5.2.).

rm
Rimuove un file (v. Sezione 3.3.2.).

rmdir
Rimuove una directory (v. Sezione 3.3.2.2.).

roff
Un compendio del sistema di typsetting di roff (v. Sezione 8.1.2.).

rpm
Gestore di pacchetti RPM (v. Sezione 7.5.2.1.).

rsh
Shell remota (non sicura!) (v. Sezione 10.4.2.).

rsync
Sincronizza due directory (v. Sezione 9.3.).

S

scp
Copia remota sicura (v. Sezione 10.4.4.4.).

screen
Gestore dello schermo con emulazione VT100 (v. Sezione 4.1.2.1.).

set
Mostra, imposta o modifica una variabile.

setterm
Imposta gli attributi di terminale.

sftp
FTP sicuro (criptato) (v. Sezione 10.4.4.1.).

sh
Apre una shell standard (v. Sezione 3.2.3.2.).

shutdown
Spegne il sistema (v. Sezione 4.2.6.).

sleep
Attende per un certo tempo (v. Sezione 4.4.1.).

slocate
Versione di avanzata sicurezza di GNU locate (v. Sezione 3.3.3.3.).

slrnn
Cliente Usenet in modalità testo (v. Sezione 10.2.6.).

snort
Strumento di scoperta delle intrusioni in rete.

sort
Riordina le linee di file di testo (v. Sezione 5.3.2.).

spell
Controllo ortografico (v. Sezione 5.1.2.3.).

ssh
Shell sicura (v. Sezione 10.4.4.).

ssh-keygen
Generazione di chiave di autenticazione, gestione e conversione (v. Sezione 10.4.4.5.).

stty
Cambia e stampa le impostazioni della linea di terminale.

su
Cambia utente (v. Sezione 3.2.1., Sezione 7.5.3.2. e Sezione 10.4.6.).

T

tac
Concatena e stampa i file al contrario (v. *cat*).

tail
Restituisce l'ultima parte dei file (v. Sezione 3.3.4.3.).

talk
Parlare con un utente.

tar
Utilità di archiviazione (v. Sezione 9.1.1.1.).

tcsh
Apre una shell Turbo C (v. Sezione 3.2.3.2.).

telinit
Inizializzazione del controllo dei processi (v. Sezione 4.2.5.).

telnet
Interfaccia utente al protocollo TELNET (non sicuro!) (v. Sezione 10.4.2.).

tex
Per formattare ed impostare tipograficamente un testo (v. Sezione 8.1.2.).

time
Temporizza un semplice comando o restituisce l'utilizzo delle risorse (v. Sezione 4.3.2.).

tin
Programma di lettura delle news (v. Sezione 10.3.6.).

top
Mostra i processi ordinati in base all'occupazione della CPU (v. Sezione 4.1.4., Sezione 4.3.5.3. e Sezione 4.3.5.4.).

touch
Cambia le marche temporali dei file (v. Sezione 7.1.2.1.).

traceroute
Stampa i pacchetti di instradamento presi dall'host di rete (v. Sezione 10.2.6.3.).

tripwire

Un verificatore dell'integrità dei file per sistemi UNIX (v. Sezione 10.4.5.).

troff
Formatta documenti (v. Sezione 8.1.2.).

tvtime
Un'applicazione televisiva di alta qualità.

twm
Tab Window Manager per il sistema X Window.

U

ulimit
Controlla le risorse (v. Sezione 7.1.2.5.).

umask
Imposta la maschera utente della creazione di file (v. Sezione 3.4.2.3.).

umount
Smonta un file system.

uncompress
Decomprime file compressi.

uniq
Rimuove le linee duplicate partendo da un file ordinato (v. Sezione 5.3.2.).

up2date
Aggiornamento dei pacchetti RPM (v. Sezione 7.5.3.3.).

update
Demone del kernel per ripulire i buffer sporchi verso il disco.

update-rc.d
Configura gli scrpt di inizializzazione (v. Sezione 4.2.5.1.).

uptime
Mostra da quanto è in funzione il sistema e il carico medio (v. Sezione 4.1.4. e Sezione 4.3.5.2.).

urpmi
Aggiorna i pacchetti RPM (v. Sezione 7.5.3.3.).

userdel

Cancella un account di utente ed i relativi file.

vi(m)
Avvia l'editor vi (o vi "improved") (v. Sezione 6.1.2.3.).

vimtutor
Il tutorial di Vim.

vmstat
Riporta le statistiche della memoria virtuale (v. Sezione 4.3.5.4.).

W

w
Mostra chi è collegato e cosa sta facendo.

wall
Invia un messaggio al terminale di ognuno (v. Sezione 4.1.6.).

wc
Stampa il numero di byte, parole e linee contenute nei file (v. Sezione 3.2.1.).

which
Mostra l'intero percorso dei comandi (di shell) (v. Sezione 3.2.1. e Sezione 3.3.3.2.).

who
Mostra chi è collegato (v. Sezione 4.1.6.).

whoami
Stampa l'effettivo ID di utente.

whois
Interroga un database whois o di soprannomi (v. Sezione 10.2.6.4.).

write
Invia un messaggio ad un altro utente (v. Sezione 4.1.6.).

X

xargs
Crea ed esegue delle linee di comando a partire da uno standard input (v. Sezione 3.3.3.3.).

xauth
Programma di utilità per i file X authority.

xawtv
Un programma X11 per guardare la TV.

xcdroast
Interfaccia grafica di cdrecord (v. Sezione 9.2.2.).

xclock
Orologio analogico/digitale per X.

xconsole
Controlla i messaggi della console di sistema con X.

xdm
X Display Manager con supporto per XDMCP, con selettore di host (v. Sezione 4.2.4. e Sezione 7.3.2.).

xdvi
Visualizzatore di DVI (v. Sezione 8.1.2.2.).

xedit
Editor grafico X Window (v. Sezione 6.3.3.3.).

xfs
Server X di font.

xhost
Programma per X di controllo degli accessi ai server (v. Sezione 10.4.3.2.).

xine
Un lettore video gratuito (v. Sezione 11.3.).

xinetd
Il demone esteso di servizi Internet (eXtended InterNET services Daemon) (v. Sezione 10.3.1.2.).

xload
Visualizzazione sotto X della media dei carichi di sistema (v. Sezione 4.3.5.6.).

xlsfonts
Visualizzatore per X dell'elenco dei server di font.

xmms
Lettore audio per X (v. Sezione 11.2.2.1.).

xpdf
Visualizzatore di PDF (v. Sezione 8.1.2.2.).

xterm
Emulatore di terminale per X.

yast
Strumento di amministrazione del sistema contenuto nel Novell Suse Linux.

yum
Aggiorna pacchetti RPM (v. Sezione 7.5.3.3.).

Z

zapping
Un visore TV per ambiente Gnome.

zcat
Comprime o espande file.

zgrep
Ricerca espressioni regolari possibilmente all'interno di file compressi.

zmore
Filtro per la visione di testi compressi.

Indice